**WIN! 10 NEW CLUSTER HATS**

# The MagPi

Issue 77 | January 2019 | magpi.cc

The official Raspberry Pi magazine

## 1970s Computing

Go back in time with the PiDP-11

## 20 Amazing Accessories

Best Raspberry Pi kit & components

# MAKE WITH CODE

**Learn Python & control electronics with Raspberry Pi**

## Plus!

- Save files in the Command Line
- The Raspberry Pi SelfieBot
- Gaming with Picade Console

## Build a Smart Door

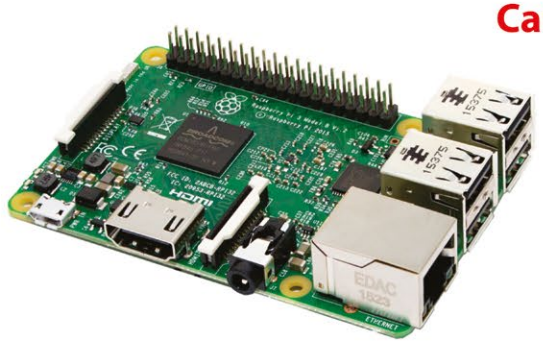
Control your locks with Raspberry Pi

## The 10 best DAC sound cards

GLOBAL DELIVERY
magpi.cc/store

Raspberry Pi PRESS

CanaKit™



# CanaKit Raspberry Pi 3 Ultimate Starter Kit

**Model B | 1 GB RAM | 1.2 GHz | Quad-Core CPU**

> Learn to Code
> Explore Computing
> Get started with Electronics

**KIT INCLUDES RASPBERRY PI 3 AND ...**

- PREMIUM CASE & HEAT SINKS
- 2.5A POWER ADAPTER
- 32 GB CLASS 10 MICROSD CARD — PRE-LOADED WITH OPERATING SYSTEM
- USB MICROSD CARD READER
- PREMIUM HDMI CABLE
- QUICK-START GUIDE
- GPIO TO BREADBOARD INTERFACE BOARD
- RIBBON CABLE
- FULL-SIZE BREADBOARD
- JUMPERS — MALE TO MALE & MALE TO FEMALE
- LEDS
- RESISTORS & PUSH-BUTTONS

Available for worldwide shipping at:

## WWW.CANAKIT.COM

## Raspberry Pi Zero W

**Now available at CanaKit!**

Wi Fi™ Bluetooth®

**ELECTRONIC KITS • ELECTRONIC PARTS • RASPBERRY PI • ARDUINO**

# WELCOME
## to The MagPi 77

It's incredibly important to make stuff. The Raspberry Pi computer, and community, is – at its heart – all about the things you can make with this wonderful low-cost machine.

Many makes are frivolous, like WhooPi cushions and Robotic Teapots; some are incredibly important, like monitoring the Borneo rainforest. Learning is important no matter what the outcome.

This issue's lead feature is Make with Code (page 26). In it you'll discover how to use the building blocks of electronics to build projects.

The Raspberry Pi can become just about anything with the right accessory: from screens, to robotic motor controllers, through to GPS and AI accelerators. Our guide to The Best Pi Accessories (page 68) is your dream toolbox.

Every issue of *The MagPi* is packed full of inspirational projects. This month we've got an e-ink display work of art (page 22), a robot that takes selfies (page 16), a webpage testing machine (page 10) and even an automated bird caller (page 20).

Build something fun. Something you love. Start with the fun stuff and you never know where it might end up. The Raspberry Pi has been used to save eyesight in India, and protect the coral reefs of Australia. But whether you build something world-changing or just mess around for fun, make sure you make stuff.

**Lucy Hattersley** Editor

EDITOR

**Lucy Hattersley**

Editor of *The MagPi*. Lucy codes, crafts, and creates wonky robots. She speaks French (badly) and mangles the piano. One day she'll get that pet dog.

magpi.cc

GET A NEW RASPBERRY PI 3A+ WITH A SUBSCRIPTION! PAGE 24

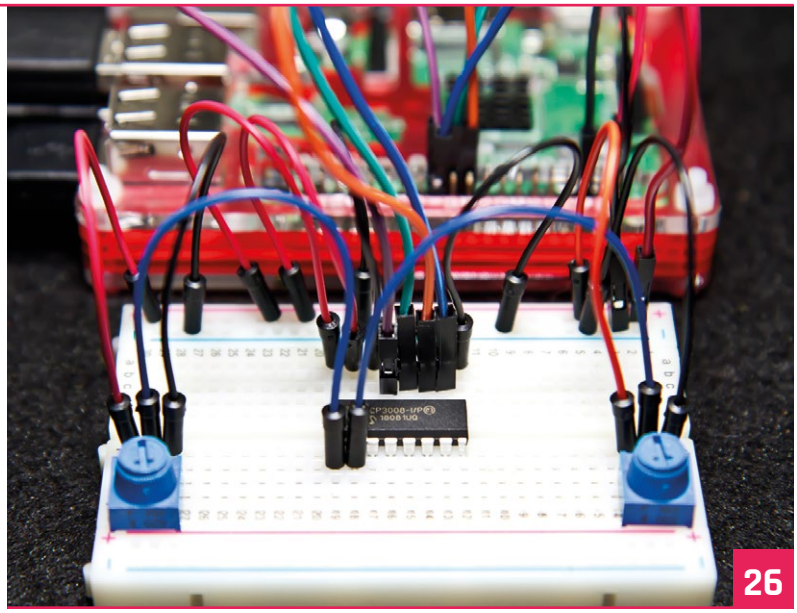
WIN! 10 NEW CLUSTER HATS
The MagPi
The official Raspberry Pi magazine
MAKE WITH CODE
Learn Python & control electronics with Raspberry Pi
The 10 best DAC sound cards

# Contents

Issue 77 > January 2019

## Cover Feature

**26** Make with code



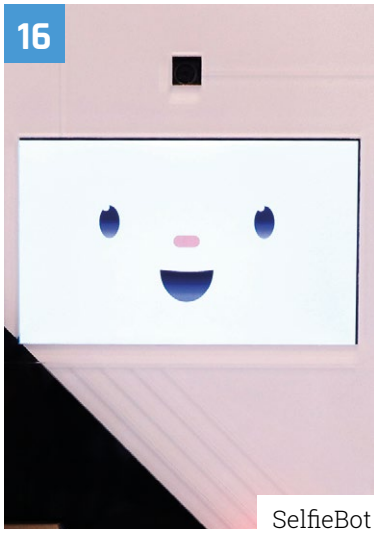

## Regulars

**06** The world of Pi
**92** Your letters
**97** Next month
**98** Final word

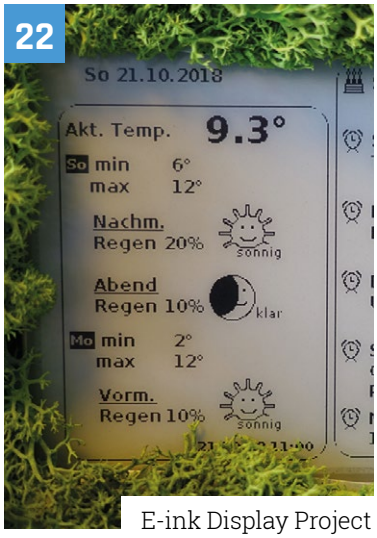
## Project Showcases

**10** WebPageTest
**14** Solar Inverter Monitor
**16** SelfieBot
**18** Open Sesamessage
**20** Game Bird Recall
**22** E-ink Display Project

WebPageTest

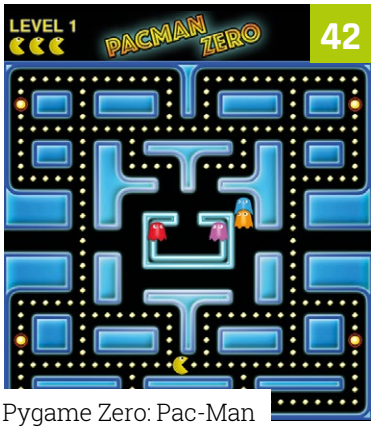



SelfieBot

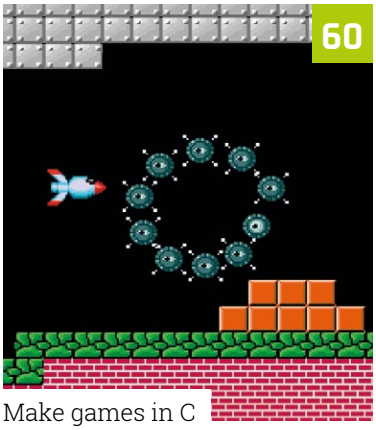



E-ink Display Project

**DISCLAIMER:** Some of the tools and techniques shown in The MagPi magazine are dangerous unless used with skill, experience, and appropriate personal protection equipment. While we attempt to guide the reader, ultimately you are responsible for your own safety and understanding the limits of yourself and your equipment. Children should be supervised. Raspberry Pi (Trading) Ltd does not accept responsibility for any injuries, damage to equipment, or costs incurred from projects, tutorials or suggestions in The MagPi magazine. Laws and regulations covering many of the topics in The MagPi magazine are different between countries, and are always subject to change. You are responsible for understanding the requirements in your jurisdiction and ensuring that you comply with them. Some manufacturers place limits on the use of their hardware which some projects or suggestions in The MagPi magazine may go beyond. It is your responsibility to understand the manufacturer's limits.

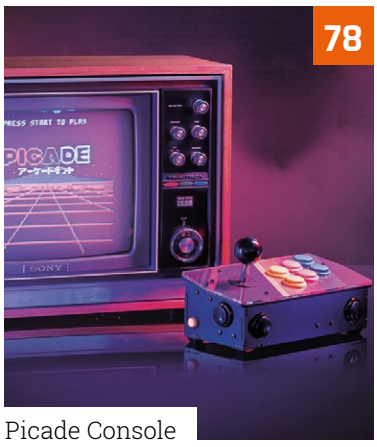


Pygame Zero: Pac-Man



Make games in C

PiDP-11



Picade Console

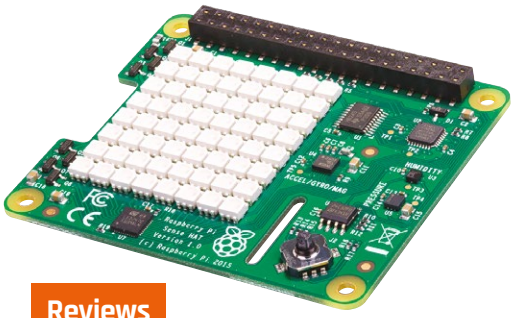
Top 10 DACs

## Tutorials


**36** Make a VDU
**42** Pygame Zero: Pac-Man part 2
**52** Smart door
**56** Command line – save it now
**60** Make games in C part 12


## The Big Feature


**68** The best Raspberry Pi accessories

## Reviews


**76** PiDP-11
**78** Picade Console
**80** Top 10 DACs
**82** Electronics resources

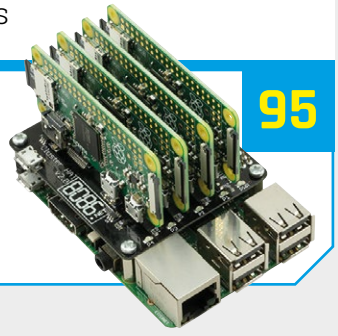

## Community

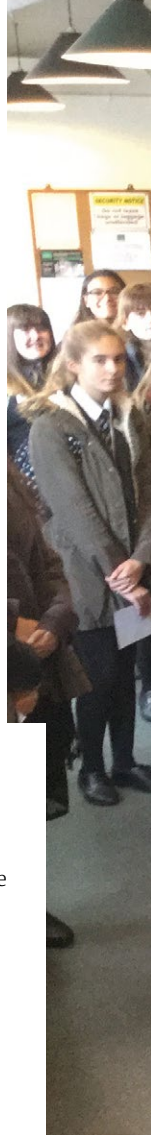

**84** Interview
**86** This month in Raspberry Pi
**90** Events


**WIN** ONE OF **TEN**

# Cluster HATs

and create a Pi powerhouse!





# Bletchley Park hosts Girls' STEM Day

National Museum of Computing holds STEM event for girls. Bravo, says **Rosie Hattersley**

Bletchley Park played host to a hugely successful Girls' STEM Day in late November, creating a real buzz among participants and demands for more events of the kind.

Among the sessions were Raspberry Pi workshops held in The National Museum of Computing's (**tnmoc.org**) newly refurbished classroom. *The MagPi* writer Mark Vanstone witnessed students getting stuck in to Python coding and Raspberry Pi-based Minecraft.

> It's no secret that there's a significant imbalance in the numbers of males and females in STEM

The female-focused STEM showcase was held at the National Museum of Computing, sited at the famous code-breakers' venue with the express intention of highlighting the breadth of career possibilities to students who may not otherwise have been aware of their aptitude for such industries. It's no secret that there's a significant imbalance in the numbers of males and females in STEM (science, technology, engineering, and maths)-based careers.

## Untapped potential

Cybersecurity company Sophos was on the scout for potential computer scientists of the future. VP Ali Kennedy noted that more than half the UK population is female and that "much more female involvement will be vital as we move into the fourth industrial revolution." Meanwhile, Buckinghamshire High Sheriff Professor Ruth Farwell gave an inspirational talk in which she

▼ Students were challenged to complete tasks using Raspberry Pi computers

▲ The Girls' STEM event was held at the National Museum of Computing, sited at the famous code-breakers' building

▼ The RAF held a STEM Engineering Challenge with students working in teams

credited her own start on the road to a high-profile career to studying a STEM subject at university. While many of the 80-plus participants on the day already had an ongoing interest in science and technology, lots of attendees were delighted to unlock their previously undiscovered talents, and impressed observers with their teamwork and ingenuity.

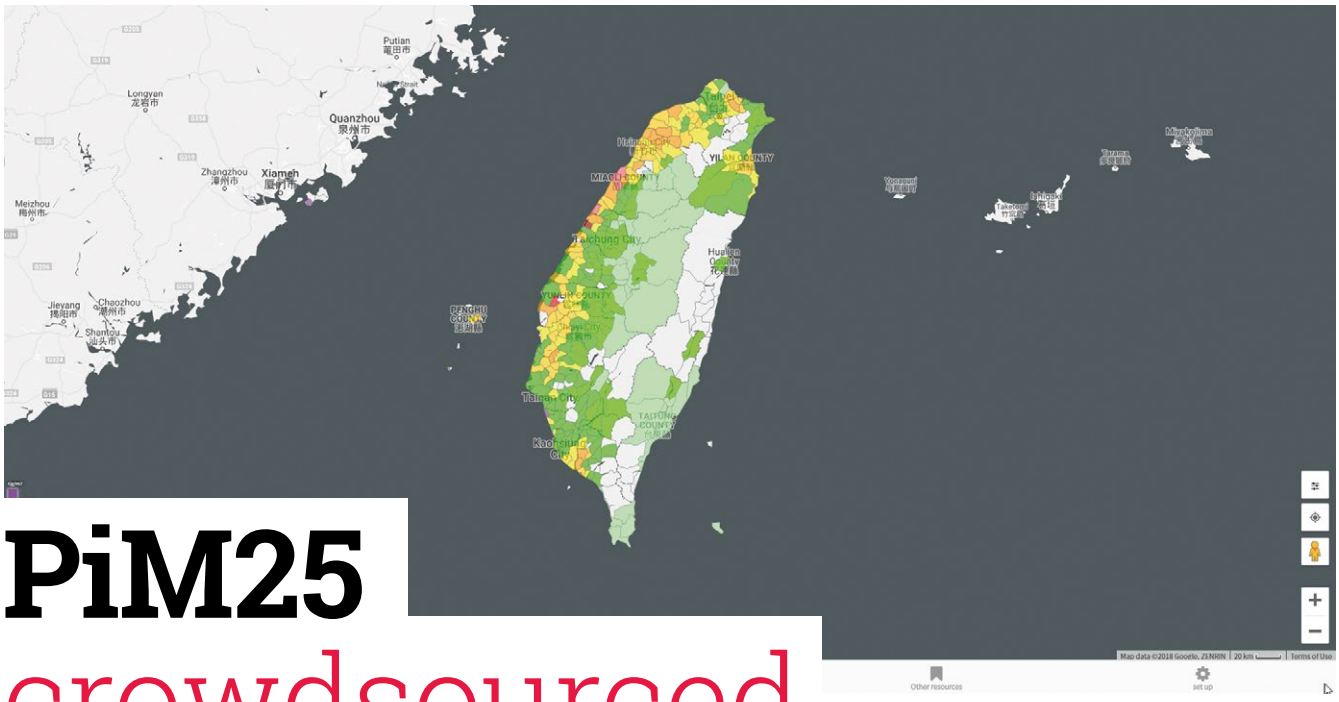
The RAF held a STEM Engineering Challenge, in which students had to master unfamiliar equipment and work in teams in order to complete tasks. Challenge organiser Chris Mossman came away seriously impressed with the students' aptitude for logical thinking and problem-solving.

PJ Evans, a TNMOC volunteer and *The MagPi* contributor commented, "More than 80 young women walked into The National Museum of Computing. I am certain more than one scientist, engineer, and programmer walked out. We saw eyes light up as the realisation dawned upon these young women that technology is as much for them and within their grasp as it is for anyone else."

## Career opportunities

Peter Membrey, a software engineer who sponsored the event, said, "I work with many problem-solvers on a daily basis, but sadly there aren't many females among them. I think lots of girls miss out on these career opportunities because currently they are seldom helped to discover that they might have a natural gift for solving problems – which is, after all, what engineering is at its core."

Event organiser Jacqui Garrad is already working on two events for 2019: one exclusively for girls, and another which will encourage teamwork among mixed teams of girls and boys.




# PiM25 crowdsourced weather data

People with Pi boards are using the PM2.5 Open Data Portal software to share weather and air pollution data in Taiwan

▲ There's not a huge amount of hardware needed for the PiM25 boxes

▼ Due to the live data update, the map may not always look this way

We like to see citizen science here at *The MagPi* – you may remember Raspberry Shake, the seismometer add-on for the Pi, which crowdsourced seismic activity. In East Asia, we hear a fair bit about issues with air quality, so we weren't too surprised when Nai-Wen, organiser of the Taiwanese Raspberry Pi community, told us about a project using open-source software to keep track of air pollution in Taiwan.

"PM2.5 is a citizen science project. We provide instructions and open-source code to help people create and customise their own PiM25 box." Nai-Wen tells us via email. "When the PiM25 box is ready, air quality and GPS data can upload to the Taiwan's PM2.5 Open Data Portal (Location Aware Sensing System, or LASS) 24/7. Everyone can see the open data from the map (**magpi.cc/rdfiGe**)."

## Mapping the air

The visualisation is fairly straightforward, colour-coding the results. Taipei, the capital, is fairly clean, but the west coast tends to be a little polluted. Data is updated on the fly, so these values may change depending on wind, time of day, and even time of year.

"We wanted to build up an easy-to-use airbox and visualise the environment information," Nai-Wen explains. "We hope the awareness will increase their interest in improving their air, health, life, and planet."

## Raspberry Pi in Taiwan

While it's sometimes easy to think of the Raspberry Pi community as being largely UK based, it's important to remember that people use it all around the world, including in Taiwan.

Starting in 2013, there have been regular meet-ups every month or two, organised by Nai-Wen: "We have workshops, hackathons, and an occasional book club event. The community also helps to organise open-source conferences, such as PyCon Taiwan or MakerConf Taiwan."

We look forward to seeing what else comes out of the Taiwanese Pi community!

# 3 ISSUES FROM £5

on a quarterly subscription

YOUR OFFICIAL RASPBERRY PI MAGAZINE
The MagPi
Issue 75 | November 2018 | magpi.cc
YOUR OFFICIAL RASPBERRY PI MAGAZINE
The MagPi
Issue 76 | December 2018 | magpi.cc
THE RASP
SUPER
• Get going guide • Starter pr
The All New 3A+
The ultimate $25 computer for makers
PLUS
10 Raspberry Pi Christmas gifts
Script command line tasks
Meet Nybble: the powerful robotic kitten
Scratch Resources | Raspberr
WIN! 10 NEW CLUSTER HATS
The MagPi
The official Raspberry Pi magazine
Issue 77 | January 2019 | magpi.cc
20 Amazing Accessories
Best Raspberry Pi kit & components
1970s Computing
Go back in time with the PiDP-11
MAKE WITH CODE
Learn Python & control electronics with Raspberry Pi
Build a Smart Door
Control your locks with Raspberry Pi
Plus!
Save files in the Command Line
The Raspberry Pi SelfieBot
Gaming with Picade Console
The 10 best DAC sound cards

**Use the code MP-SAVE at checkout**

**Visit:** magpi.cc/345

# WebPageTest

Testing your website on different browsers? You may be using a Raspberry Pi server farm you never knew existed. **Rob Zwetsloot** uncovers the details

MAKER

**Patrick Meenan**

A software engineer at Cloudflare who specialises in web performance. He uses Pi boards in a lot of his testing.

**webpagetest.org**

One of the most invaluable tools to a web developer is the ability to quickly test your current build on a variety of browsers. Chrome, Firefox, Safari, Internet Explorer (unfortunately), and even mobile browsers need to be tested for. Fortunately, there are a lot of online services that can test it for you. One of these is WebPageTest (**webpagetest.org**), which just so happens to use a lot of Raspberry Pi boards.

"I largely use the Raspberry Pis as POE-powered computers to control smartphones for testing mobile web performance," its creator Patrick Meenan tells us. "The public instance of WebPageTest has upwards of 80 Raspberry Pis driving the mobile testing for all of the Android and iOS devices."

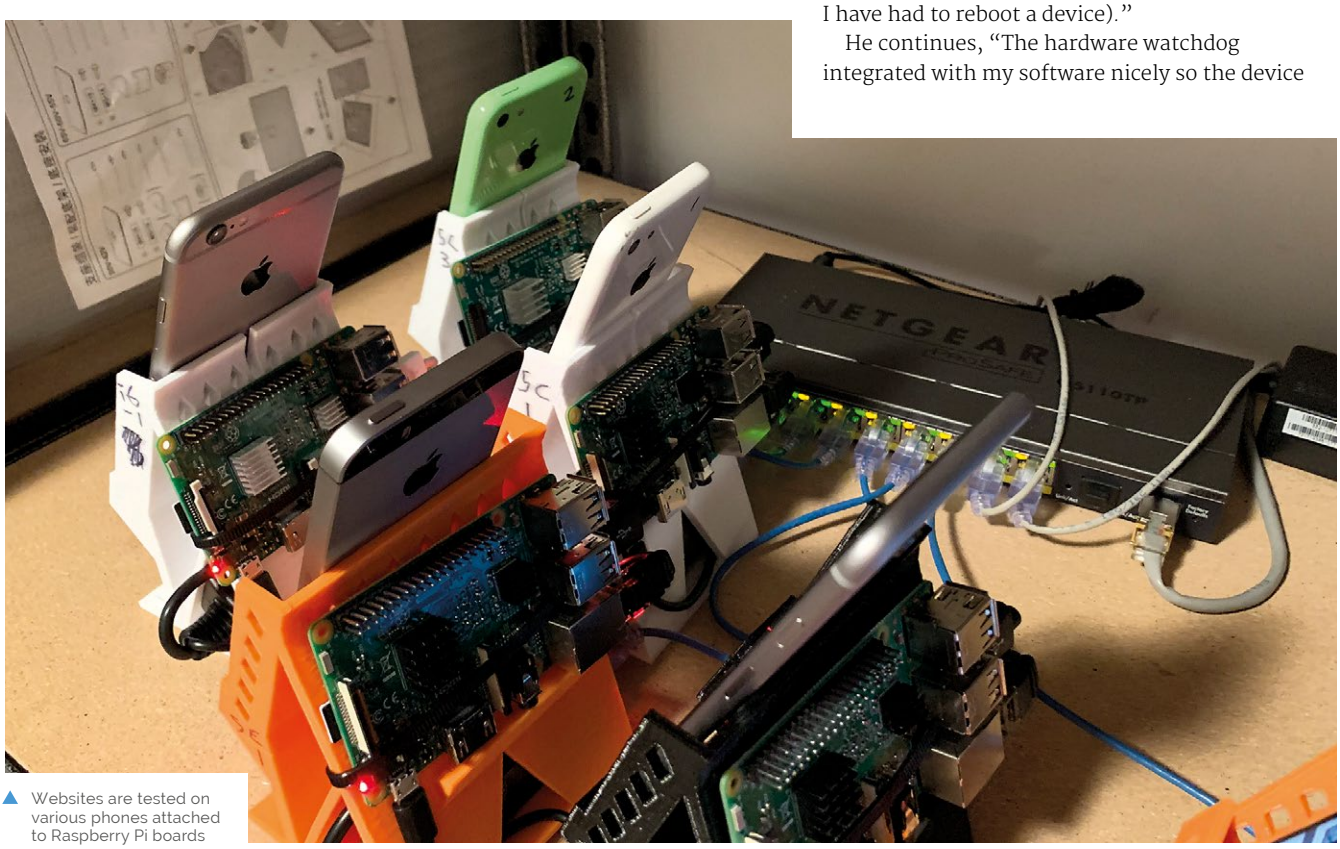
Patrick began using a similar setup while he was at Google working on web performance. Since then, he's built it up and deployed it under the WebPageTest website, and the Raspberry Pi was the only option for it.

## Perfect with Pi

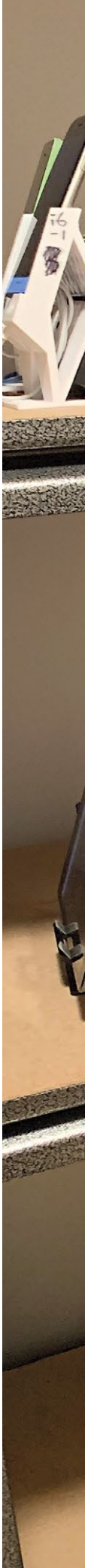
"There are a ton of things that make them perfect for my use case," Patrick explains. "They are inexpensive enough that I can use a dedicated Raspberry Pi for each phone or tablet in the lab.

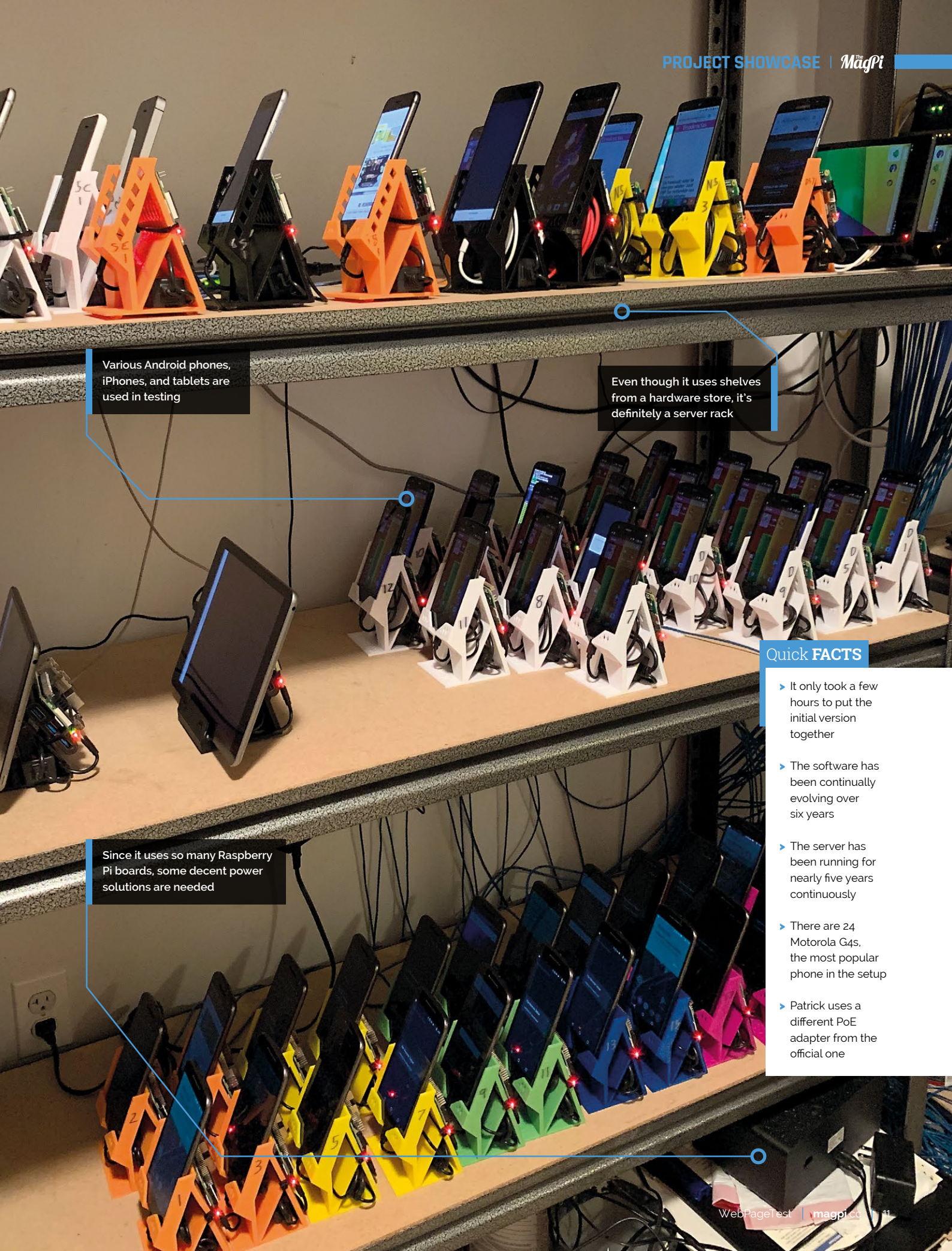
"They provide enough power over their USB port to both power and control most devices. They can be powered over PoE, which gives me a way to remotely hard-reboot a device if needed (used to be a regular issue, but it has been months since I have had to reboot a device)."

He continues, "The hardware watchdog integrated with my software nicely so the device




▲ Websites are tested on various phones attached to Raspberry Pi boards





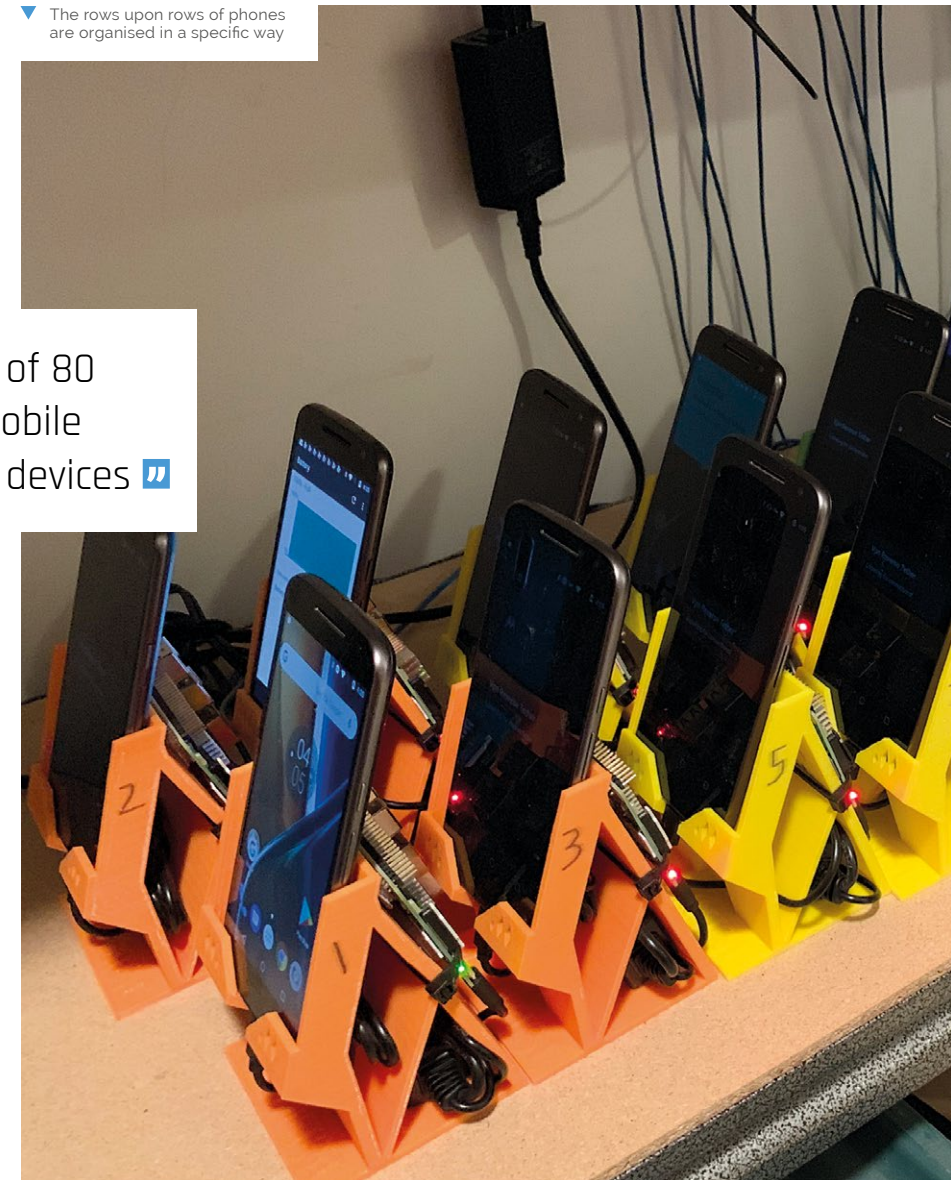
Various Android phones, iPhones, and tablets are used in testing

Even though it uses shelves from a hardware store, it's definitely a server rack

Since it uses so many Raspberry Pi boards, some decent power solutions are needed

## Quick **FACTS**

- It only took a few hours to put the initial version together
- The software has been continually evolving over six years
- The server has been running for nearly five years continuously
- There are 24 Motorola G4s, the most popular phone in the setup
- Patrick uses a different PoE adapter from the official one



The rows upon rows of phones are organised in a specific way

self-reboots automatically if anything goes catastrophically wrong.

"The software and community support are unmatched. I tried other (faster) SBCs, but they all had issues with missing packages or lack of community support."

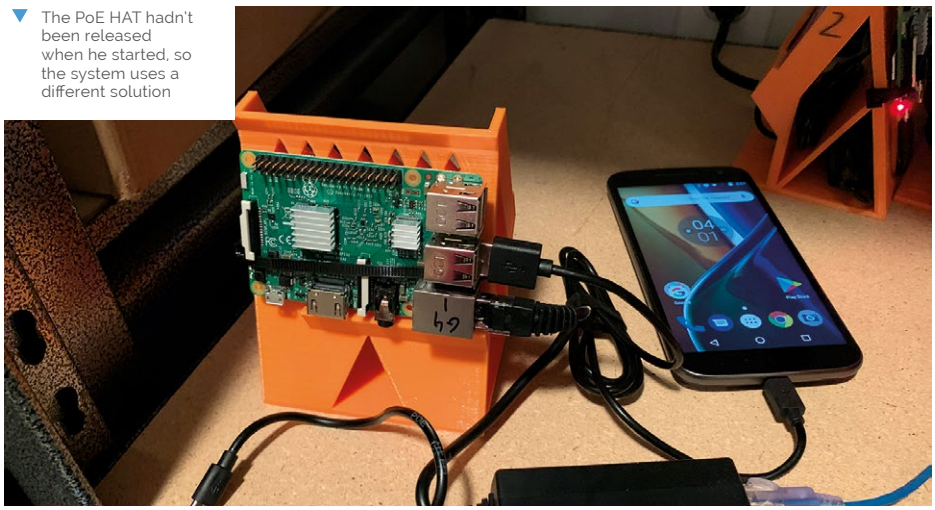
> WebPageTest has upwards of 80 Raspberry Pis driving the mobile testing for Android and iOS devices

## Simple and sturdy

It's a very simple setup beyond the software as well, merely using a PoE HAT on a Pi 3B+ with 3D-printed cases. All the individual phones are hooked up to the Pi so they can perform their testing duties.

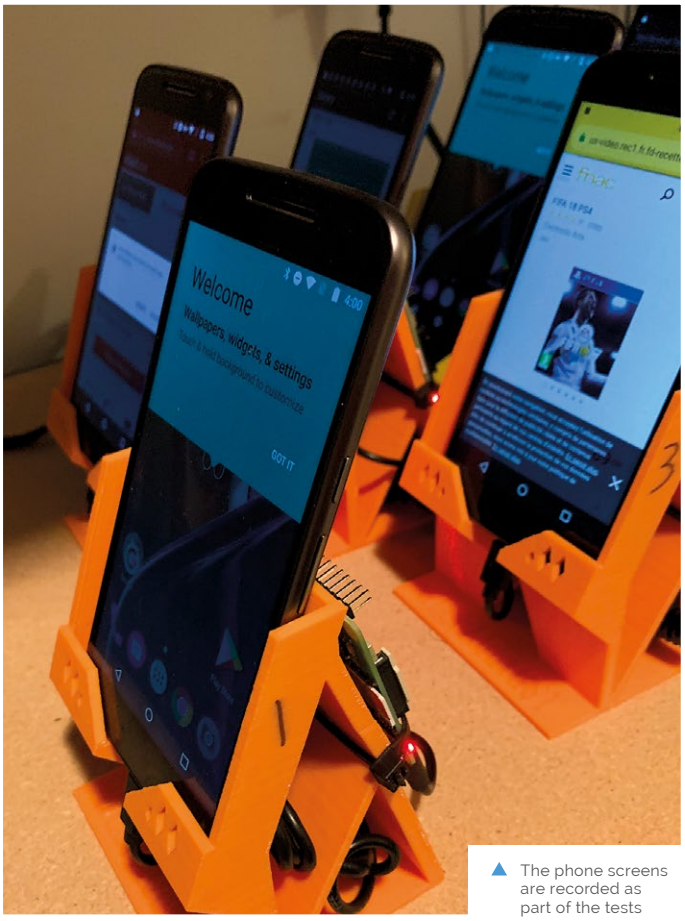
"They work *amazingly* well," Patrick says. "It has gotten to the point where the mobile device testing is as stable as the desktop testing (which runs in VMs [virtual machines]). These days the mobile devices are set-and-forget. The software deployment is now fully scripted and I have a base image that I can clone to new SD cards and have a new device up and running in a few minutes."

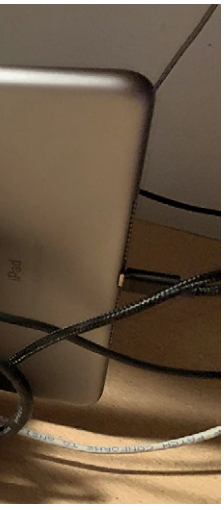
For the moment, Patrick's little Pi farm is holding up well, and is only occasionally maintained via the addition of more powerful models of Raspberry Pi when they become available. He hasn't even lost an SD card so far – a true testament to the Raspberry Pi, SD cards, and Patrick's smart coding.



The PoE HAT hadn't been released when he started, so the system uses a different solution

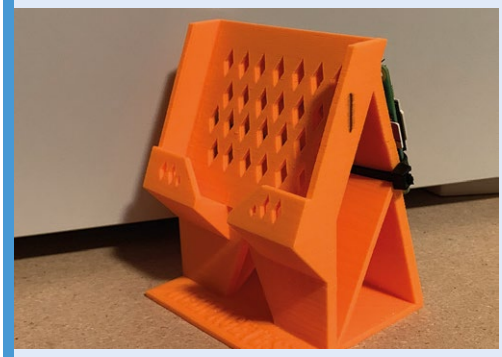
Tablets don't quite fit as well in the cases



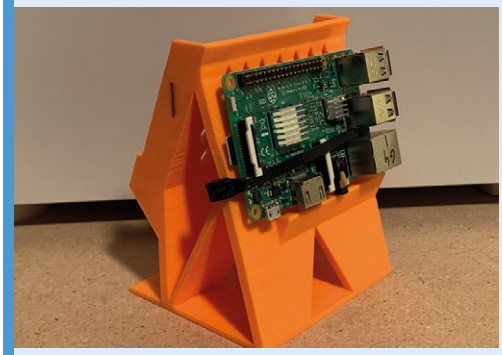



▲ The phone screens are recorded as part of the tests

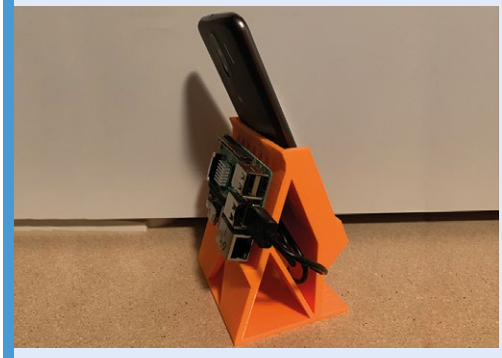
## Build a tester



**01** Not many cases you can buy are designed for connecting a Pi and a phone together – in fact, we'd venture a guess there are none. Patrick 3D-prints his own custom cases that make organising his server a bit easier.



**02** Patrick always uses the most powerful version of the Pi, as a lot of screen recording occurs in the process and a more powerful CPU is better for that. Here's a Raspberry Pi 3 strapped to the back of the custom case for just that purpose.



**03** The selected phone is powered and controlled by the Pi. The PoE HAT then powers the entire thing, and allows the Pi to connect to the network and do all the testing it needs to do.

# Solar Inverter Monitor

What do you do when it's tricky to get data from your solar power setup? Hack it with a Raspberry Pi of course, as **Rob Zwetsloot** finds out

MAKER

**Clive Maynard**

A retired software engineer and computing lecturer who has made many contributions to education.

clivemaynard.org

Renewable energy is something we're pretty interested in here on *The MagPi* – and not just because of the ways you can use it to power your outdoor Pi projects, even if that is very cool. So when Clive Maynard contacted us to share his project about keeping an eye on the performance of solar power inverters, we were all ears.

"It starts with establishing communications with the inverter and then goes on to logging the data retrieved," Clive tells us. "This involves the creation of CRC16 calculations to determine the validity of the data. Having logged the data, there is a need to display it for easy human understanding. The project involves two types of plot: the first traces the output of the solar inverter over a day of generation, and the second is a bar chart summary of the daily performance plotted over a month period."

Solar inverters take the photovoltaic (PV) output from a solar panel and convert it to a more usable AC current. Why did Clive feel the need to build this system, though? Well, his solar inverter didn't make it easy to read its data.

## A simple solution

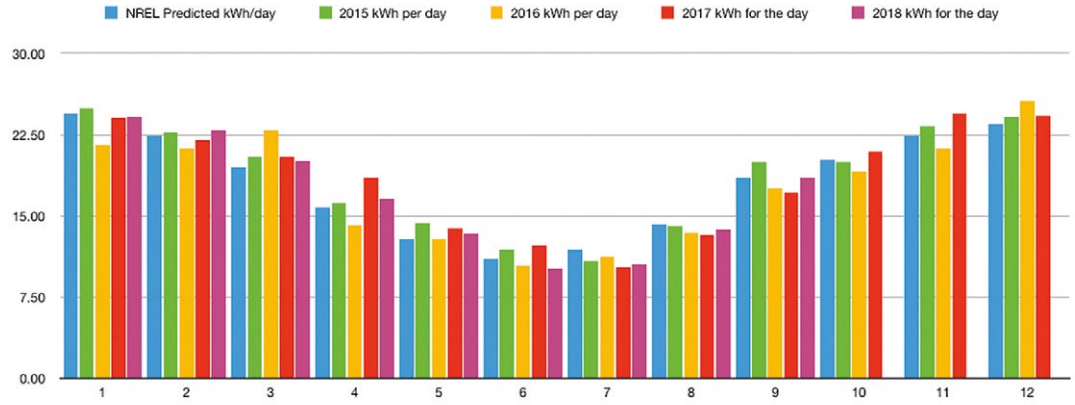
"I have had a PV system at my home since 2015 and have kept records of its performance compared with the National Renewable Energy Laboratory (NREL) predictions for my location since getting it, every month scrolling through the menus on the inverter front panel to find the information I need and entering it into a spreadsheet. Interesting, but not as good as extracting the information from the solar inverter electronically."

To read the data, all Clive needed to do was connect the RS-485 serial output to the Pi using a USB converter and then parse that output for the relevant information for his graphs.

"It is a cheap, powerful system which has a good programming environment and operating system, together with relatively easy interfaces to collect the data," explains Clive about his reasons for using a Raspberry Pi. "I have spent many years using UNIX, C, and Python, so [it was] an easy choice."

## Renewable graphing

The results speak for themselves, with graphs aplenty for data lovers to look over. It's important to know how solar energy works if we're to rely on it more in the future. For now, Clive has a couple of things he'd like to improve with his setup: "Physically I should make a more secure mounting for the Raspberry Pi and interface in my garage. I'll also set the logging program up to automatically run on reboot, to allow for power failure and get the logging back on as soon as possible."




▶ Are the national predictions accurate? Sort of

Not your standard serial output, Clive needed a USB converter to access it

Solar power needs to be converted from a variable DC output to usable AC

Data can be read directly on the box, but it takes a bit of digging

## Quick **FACTS**

- Photovoltaic current exists as DC (direct current)
- You can use **pvwatts.nrel.gov** to see your home's projected output
- Solar panels are a lot cheaper and affordable these days
- Clive has been programming since 1965
- While the output is in C, a lot of the Pi code is in Python

# SelfieBot

Want to snap a selfie and put a smile on the face of the camera? **David Crookes** meets SelfieBot

MAKER

**Sophy Wong**

Sophy is an accomplished maker as well as a designer of fashion, jewellery, costumes, and wearable technology. She has a strong background in graphic systems and marketing.

**magpi.cc/hhoBck**

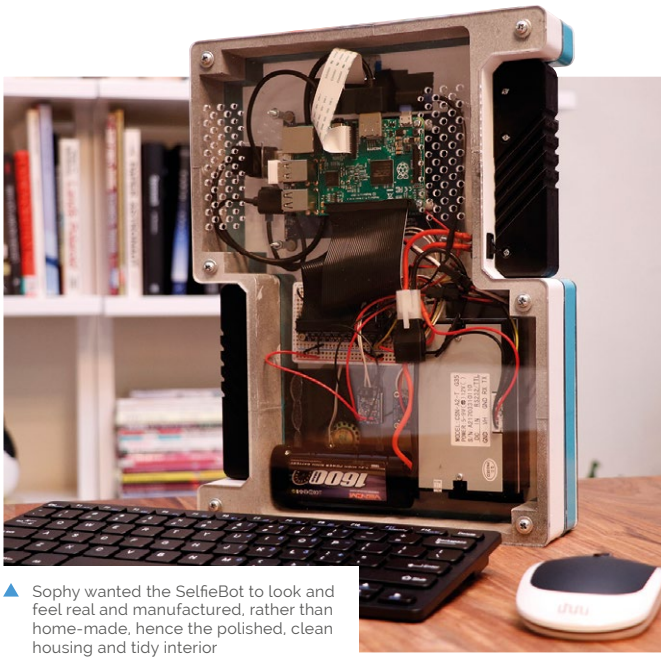
After Sophy Wong's SelfieBot wakes from a snooze, it makes a cute noise, sings a song, winks at a passer-by, and then blinks silently. Or at least, it does if it's left alone for long enough: its ability to capture fun photos and print them on to thermal paper ensures it's nearly always in demand.

SelfieBot sprung into life in 2017 when Sophy and her husband Kim Pimmel created an early version for their local Seattle Mini Maker Faire. "We thought that if we created a fun selfie-taking experience, with something visitors could take home like a printed photo, it would encourage people to the booth and give them another way to interact with our projects."

The plan worked a treat. SelfieBot – mains-powered and made from laminated laser-cut plywood – went down a storm, so the couple decided to refine the concept for 2018. Rather than have it tethered to one location, they opted for a mobile version.

"I redesigned the circuit so that it could be powered by a small 7.2V NiMH battery like the ones in radio-controlled cars," Sophy explains. The idea of putting a Raspberry Pi 2 at its heart remained, however.

The Pi is connected to a Pi Cobbler which routes all of the Pi's GPIO pins to a Perma-Proto board. A Pi Camera Module V2 is used to take photos, while a mini metal speaker is hooked up to an Adafruit mono 2.5W class D audio amplifier, to allow the SelfieBot's sounds and jokes to be clearly heard.



Sophy wanted the SelfieBot to look and feel real and manufactured, rather than home-made, hence the polished, clean housing and tidy interior

For its face, which beams bright from a 5-inch HDMI display, the couple conducted a lot of research. "We analysed the features of cute robots and scary robots, knowing that we wanted a face and sounds that were appealing, cute, and non-threatening," Sophy says.

> "We hardly print anything these days, especially photos, so it's a lot of fun"

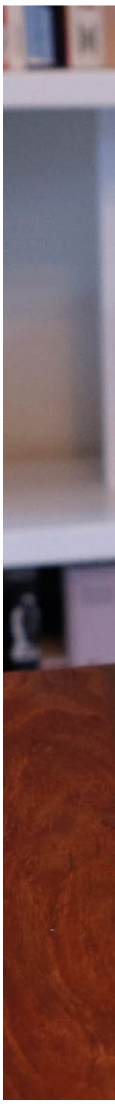
Kim created the code for this, producing animated faces – made up of individual images played in sequence. "When paired with sounds, which are recordings of my voice, the illusion is complete. We used the Pygame engine to drive the graphics and sounds, and CUPS for printing."

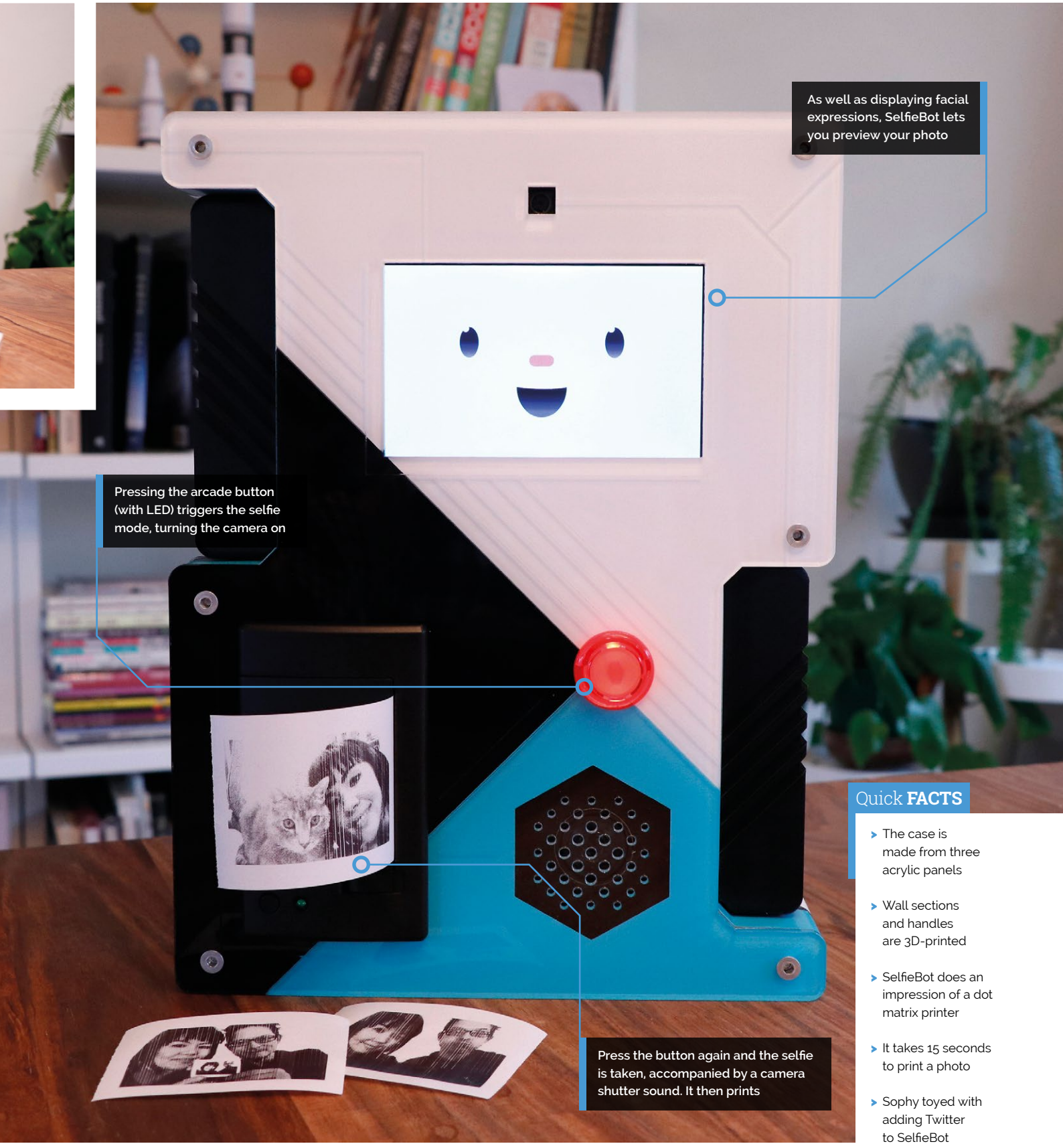
## And action

Photos are outputted to a portable thermal printer. "It's very satisfying to watch the image print before your very eyes," Sophy says. "It seems like we hardly print anything these days, especially photos, so it's actually a lot of fun for people playing with SelfieBot."

She says folks love tearing off and keeping their photo, leaving with as big a smile as that on SelfieBot itself: "When a photo is printing, SelfieBot makes a cute face which is intended as a second reward... It feels like a big win."

When the accelerometer detects it has been placed on its back, the SelfieBot falls asleep and shows a snoring animation




As well as displaying facial expressions, SelfieBot lets you preview your photo
Pressing the arcade button (with LED) triggers the selfie mode, turning the camera on
Press the button again and the selfie is taken, accompanied by a camera shutter sound. It then prints

## Quick **FACTS**

- The case is made from three acrylic panels
- Wall sections and handles are 3D-printed
- SelfieBot does an impression of a dot matrix printer
- It takes 15 seconds to print a photo
- Sophy toyed with adding Twitter to SelfieBot

# OpenSesamessage

With a simple text message, you can open Conor Breen's door. As long as you're on the guest list. **Rob Zwetsloot** takes a look

MAKER

**Conor Breen**

A computer science graduate who specialises in IoT and web development. He enjoys playing around with a Raspberry Pi in his spare time.

magpi.cc/fzzAZL

As technology and interconnectivity steam ahead, new solutions to old problems sometimes present themselves. While a trusty key may not be quite the thing of the past yet, electronic locks using RFID and other kinds of digital authentication are already occupying that space nicely. This is where OpenSesamessage comes in.

"My project is a smart home entry system designed to be modular, cheap, and cross-platform," creator Conor Breen tells us. "It allows a person to add this device to their front door and, through the website, grant people access to the device through their phone number."

While this is a little more involved than using a phone's built-in NFC capability, it probably makes it a little more secure in the process. In fact, the system makes use of Telegram, an open-source messaging service well known for its security.

## Hidden origins

"The idea stemmed from me always forgetting my keys," Conor admits. "But, because of forgetting my keys, I would invariably have my phone to call someone to either let me in or lend me theirs. So why not cut out the middleman and [let] my door be the one I text?

"It worked better than expected. The device was able to parse incoming messages; even when bombarded, it would just queue them. Users would be returned messages either allowing them entry or disallowing them. I was approached by people who were thinking up new use cases for this device; one in particular was a person who had a few Airbnb apartments, and they thought this would be a great alternative to a key box outside the apartment."

> Users would be returned messages either allowing them entry or disallowing them

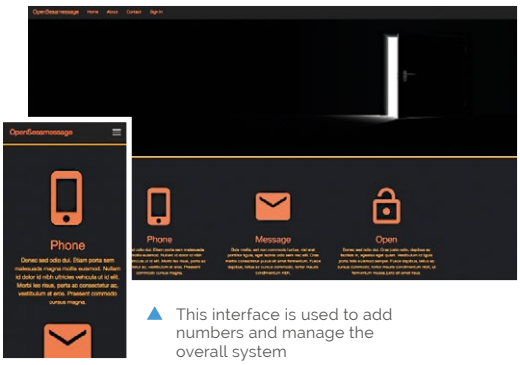
## Practical applications

While the project was tested on a model door, Conor assures us it would work just fine on a normal one – the latch lock in particular was one that you would use on a regular door anyway. Even then, there are some ways in which he'd like to improve it.

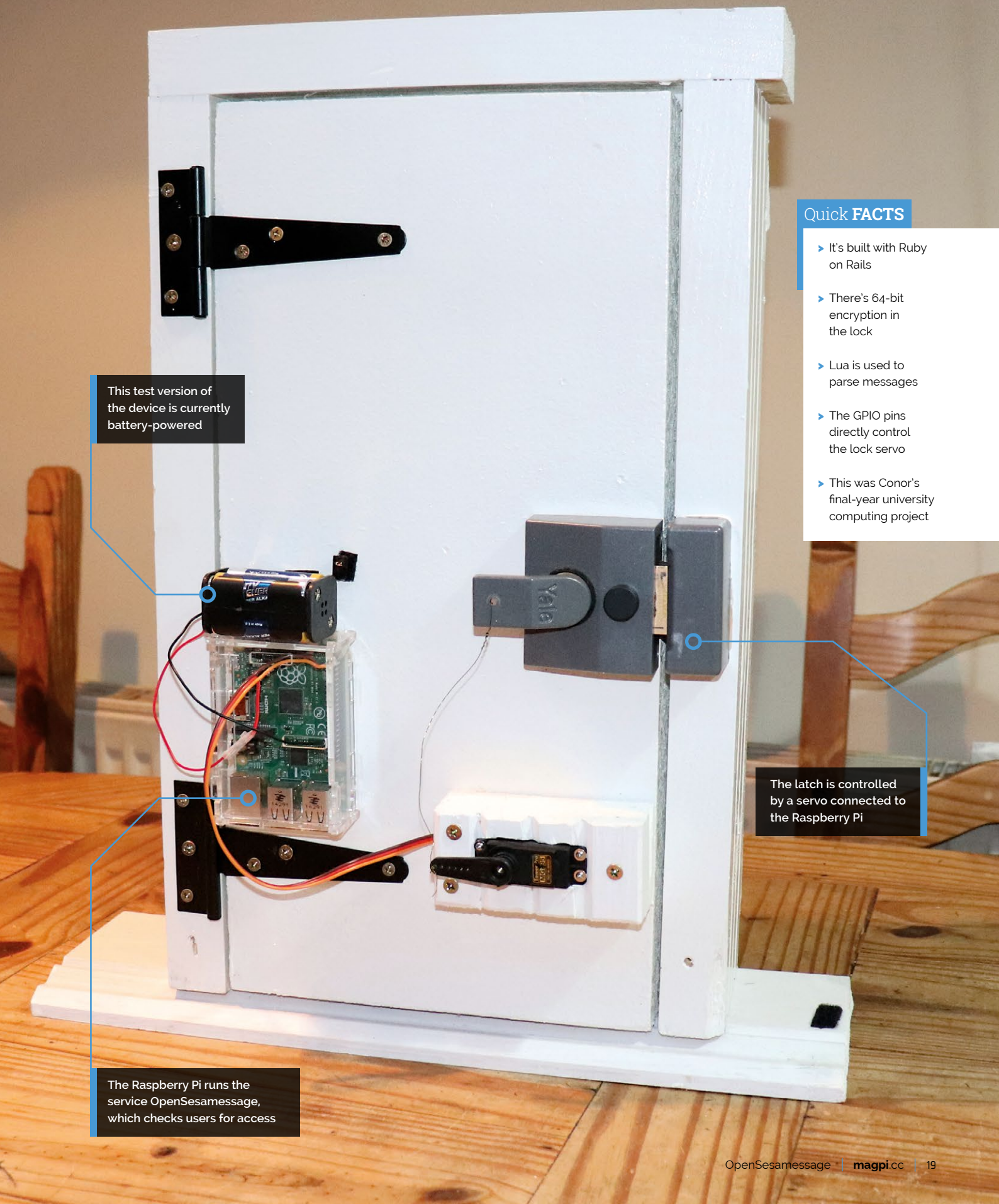
"I am interested in adding extra functionality, such as adding cron jobs to give a user privileges for a specified amount of time and when that time is up, their access would be revoked," Conor explains. "And if I had an untold amount of time, I would love to have gotten into writing a messenger service from the ground up, and have a bespoke app for the device. Hardware-wise, I would have updated the battery pack that operated the servo to be rechargeable, and maybe charged out of one of the USB ports."

You can read Conor's complete paper, full of all the technical information you'd ever need on the project, here: **magpi.cc/fzzAZL**.

▲ Conor built a small door for his project, although it uses a full-size lock




▲ This interface is used to add numbers and manage the overall system



This test version of the device is currently battery-powered

The latch is controlled by a servo connected to the Raspberry Pi

The Raspberry Pi runs the service OpenSesamessage, which checks users for access

## Quick **FACTS**

- It's built with Ruby on Rails
- There's 64-bit encryption in the lock
- Lua is used to parse messages
- The GPIO pins directly control the lock servo
- This was Conor's final-year university computing project

MAKER

**Martha Zimet**

Martha lives on her ranch in rural Nevada after more than three decades in Silicon Valley as a software engineer/architect. She now trains gun-dogs and watches over her herd of equine.

manray.com

# Game Bird Recall

Managing wildlife populations by mimicking their calls is another job for the Raspberry Pi. **Phil King** heeds the call of the wild

Requiring an active population of wild game birds to train her gun-dogs, Martha Zimet has developed a Raspberry Pi-based system to mimic the birds' rallying calls in order to 'anchor' them to a geographic area (**magpi.cc/nHzNWA**).

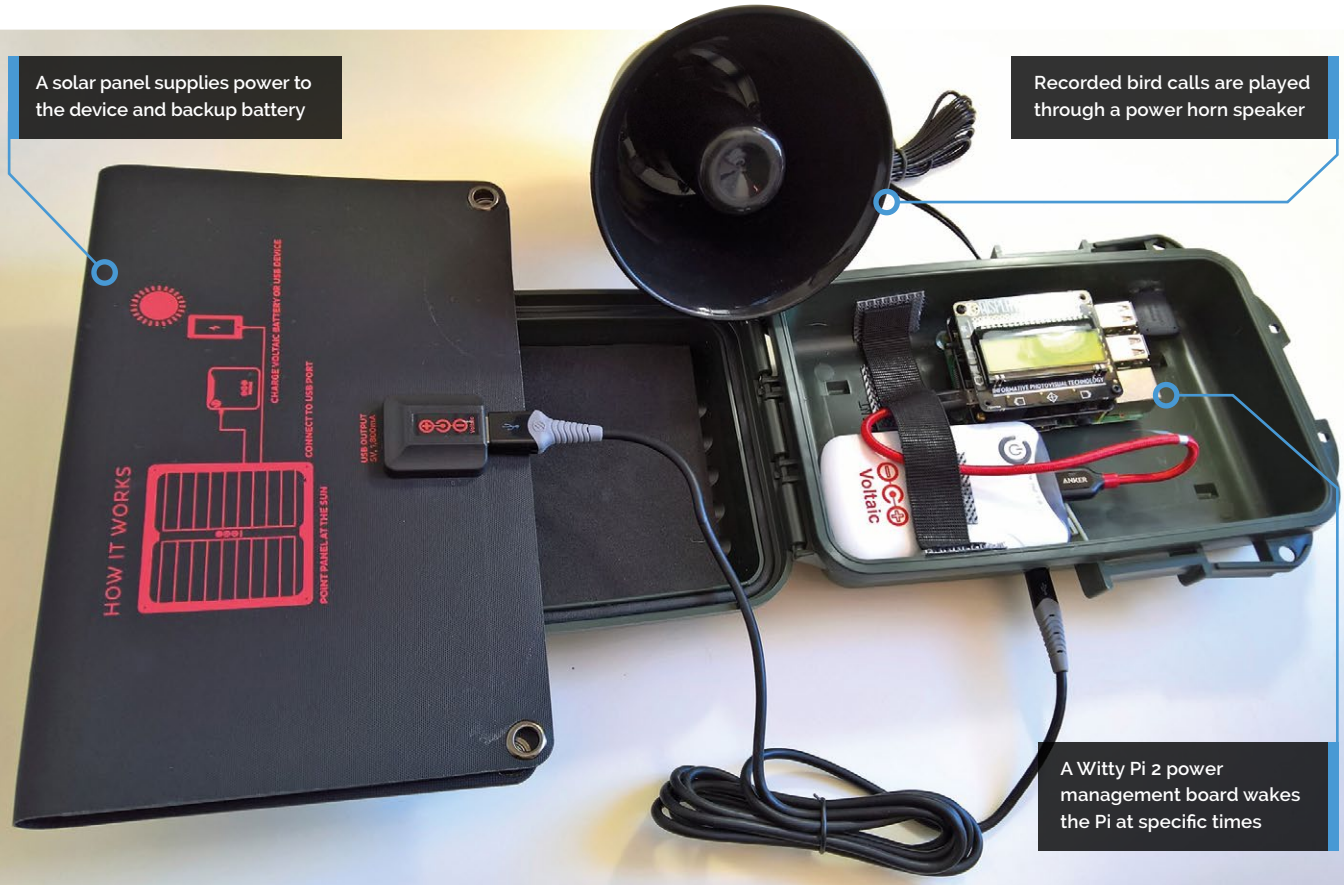
"In my experience, game birds really train gun-dogs," Martha advises. "They teach the dogs to be stealthy, not jump in on the bird, manage their own energy, otherwise, the dog never gets what it desires: the bird in its mouth. Therefore, to train your gun-dogs, what is required is 'wild acting' game birds."

Upon introducing a population of Coturnix (common) quail to her 80-acre Nevada ranch, she developed her own bird recall system to lure them to a specific location. "I do this by providing predator-proof food and water, and a 'rally call' for the specific species that effectively calls them back to the location where the food and water are located," she explains. "The released game birds become feral in about a month and act like wild game birds."

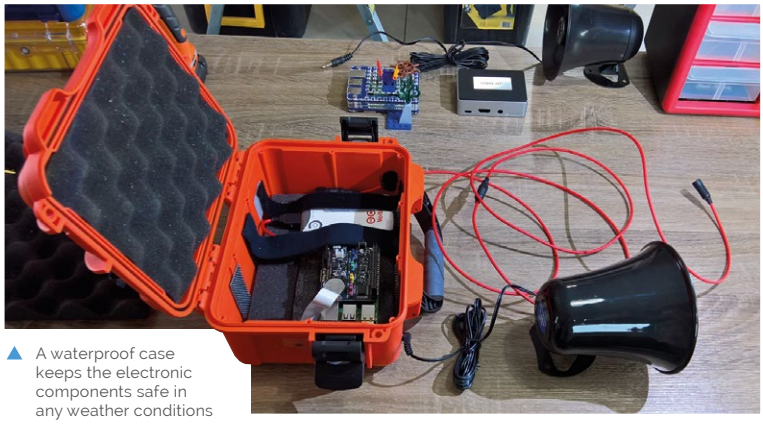
## Caught on camera

Given the system's remote location, low energy consumption is a key factor. A solar panel hooked up to a backup battery provides the electricity, while a Witty Pi 2 board handles power management. "[It] has a real-time clock based on NTP, and it runs 24/7," says Martha. "It has programmed events, which execute only during daylight hours, that play the recorded game bird call at specific intervals.

Images courtesy of Martha Zimet, CC BY-NC-SA 3.0


A solar panel supplies power to the device and backup battery
Recorded bird calls are played through a power horn speaker
A Witty Pi 2 power management board wakes the Pi at specific times
HOW IT WORKS
Voltaic

▲ Once installed out in the field, the system is self-sufficient, minimising human intervention that might affect the wildness of the birds



▲ A waterproof case keeps the electronic components safe in any weather conditions

During these events, the [Pi NoIR] camera is also active and can record any movement detected and transmit the results to me."

For the latter, Martha is using the Soracom cellular IoT network. "Since I am not transmitting live video from the devices, I can simply use 3G connectivity. It has proven to be very reliable.

> They can be used for any wild game population, and not just game birds

I use an external antenna to the GSM modem, and everything is enclosed in a waterproof case. I spent well over a month attempting to use WiFi, and even with clear line of sight to my location, high power Yagi antennas were just too wonky."

## Stickler for quality

The whole system took Martha about six months to develop. "A major portion of the time was spent validating components, where the most challenging ones were solar power, batteries, and connectivity. Since the units are installed at remote locations, they just have to work. Given my background, I am a stickler for quality. The [Python] software I wrote – my 'secret sauce' – was small in comparison to the validation process."

With a US patent pending, Martha is offering a commercial service to install similar devices in other locations. "Each system I develop is based on user requirements and use cases. So far, there is no way to mass-produce them, but they can be used for any wild game population, and not just game birds."

Martha has been invited to the Yukon next autumn, to evaluate the system for use with the wild moose population. In addition, she reveals: "I have connections in Finland and I have hopes the system could be used by the Sami population with their reindeer herds."

### Quick FACTS

- It can play up to six recorded calls
- These are selected via capacitive touch buttons
- Extra environmental sensors can be added
- A Pi NoIR camera is triggered by motion detection
- Images are transmitted via a 3G network

# E-ink Display Project

Green in more than one sense, the E-ink Display Project is both a useful tool and a rather fetching piece of art. **Nicola King** admires this ingenious piece of creativity

MAKER

**Anke Dietzen**

An enthusiastic maker in her precious spare time, Germany-based Anke Dietzen loves to learn, tinker, and create.

ankesreich.de

Did you know that e-ink displays can retain what they're showing, even without electricity? By anyone's standards, that makes them pretty ecological. The E-ink Display Project, developed by Anke Dietzen, is a resourceful and inventive take on a very cool way of displaying information.

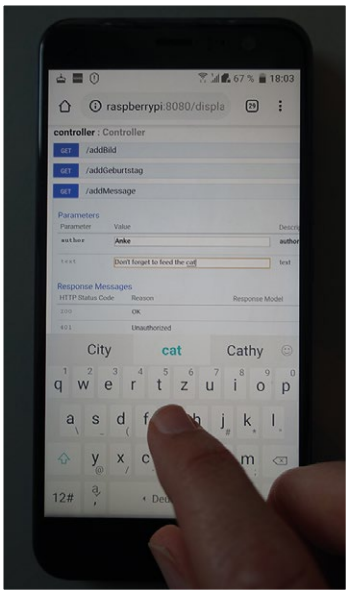
The project's origins came through reading an article about the Raspberry Pi, as Anke explains: "I thought 'how cool is that?', and I wanted to do something with the Raspberry Pi. I ordered [one] and, very enthusiastically, a big sensor kit. As I had no knowledge of electrical engineering, I had to learn some basics and I was very happy then to light a diode on a breadboard."

At the same time, Anke bought a small Waveshare e-ink display, which she favoured for its very clear and precise visuals. Armed with her new purchases, she set about coming up with a practical display that would also look good. "I wanted to do something with e-ink displays which I thought should be at least a little useful," she tells us. "As, in the morning, I normally check the weather via smartphone, I thought it would be nice to read the latest weather information and calendar entries / birthdays with just one look."

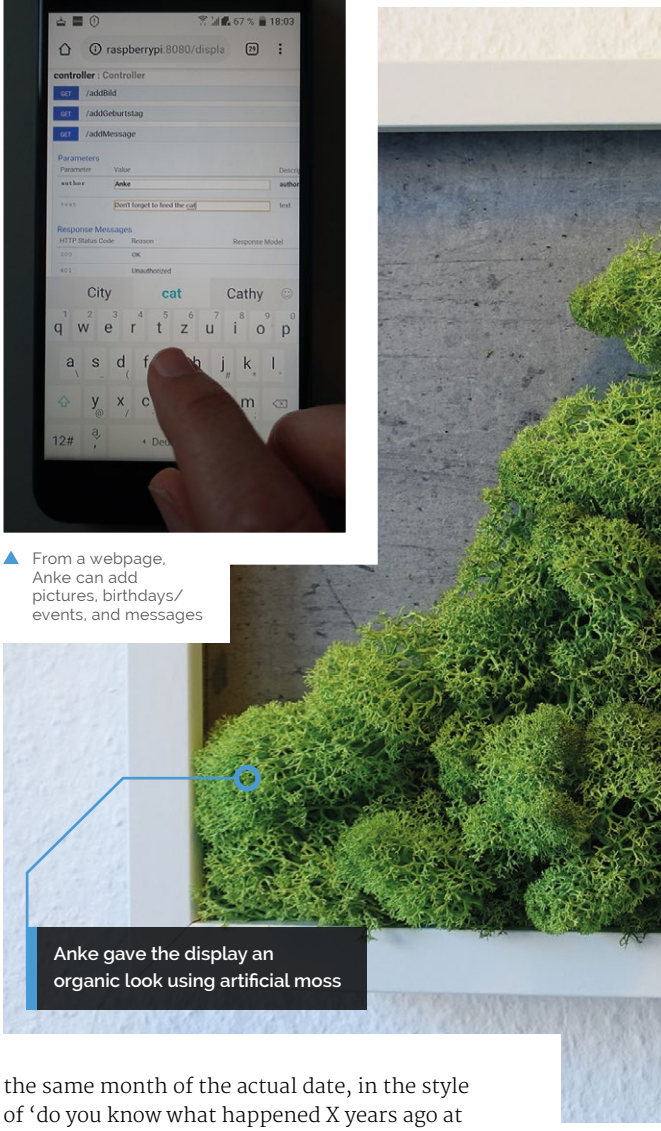
> "I had no knowledge of electrical engineering – I had to learn some basics, and was very happy to light a diode on a breadboard"

## Picture perfect

As well as including informative data such as forecasts and reminders, Anke wanted the possibility of having images on the screen, so, "I used a bigger three-colour display (7.5-inch – 640×384 pixels) for showing pictures. The picture shown was always taken in the same month of the actual date, in the style of 'do you know what happened X years ago at this time?'" she says. In addition, she describes the key benefit of using an e-ink display: "If you disconnect [it] from power, the Raspberry Pi will keep the last picture forever, unlike the 'normal' displays which need power to show something. The power consumption is very low in standby mode."



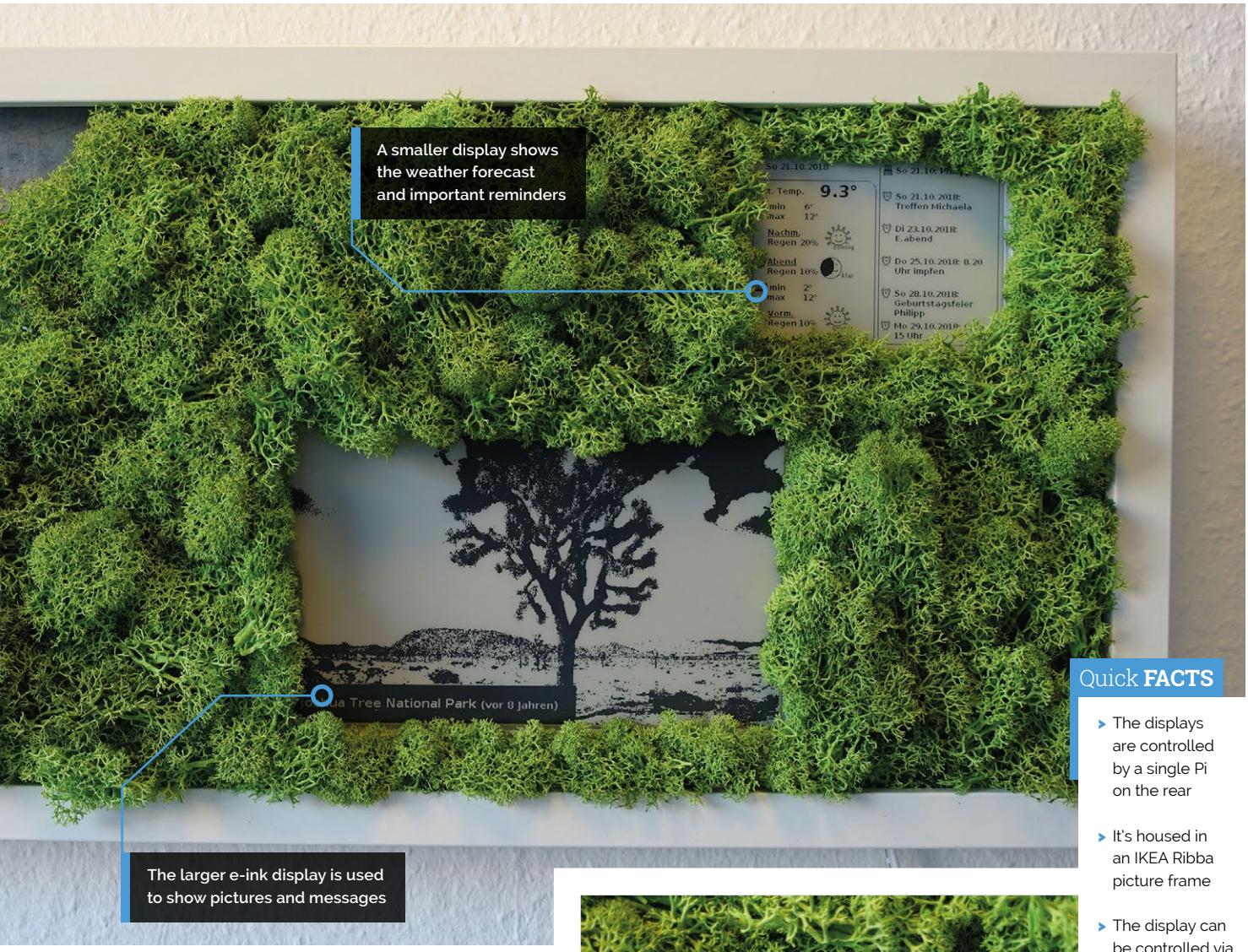

▲ From a webpage, Anke can add pictures, birthdays/events, and messages



Anke gave the display an organic look using artificial moss

## Ink...redible

So, how does it work? Anke reveals, "The Raspberry Pi controls two e-ink displays. On the Pi, all the time an application server with a database is running. Over a webpage (REST interface), the actions can be triggered: actions are, for example, 'store message in database', 'fetch weather information from internet and store in database', 'store picture info in database', 'refresh display from database', etc. For displaying a picture, or the weather information, the

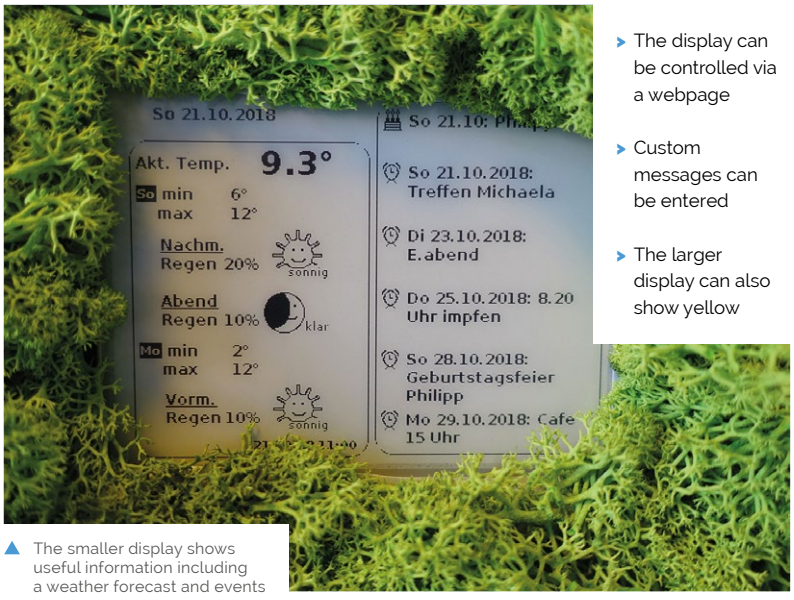

A smaller display shows the weather forecast and important reminders
The larger e-ink display is used to show pictures and messages

application creates a temporary monochrome bitmap which is then shown."

Being familiar with Java, it was natural for Anke to use it to code the project. "I first had to transfer the Python driver logic provided by the e-ink display producer to Java." She then spent some time figuring out how to get the Raspberry Pi to control two e-ink displays at the same time.

Aesthetically, the finished project is calming to the eye. Anke used some artificial moss to give it a very organic feel and, with it hanging in the kitchen of her home, the feedback that she has received from other members of the household has been, unsurprisingly, very positive.

Charming, functional, and eco-friendly, it makes a refreshing change from the more mundane pinboard or sticky note!




▲ The smaller display shows useful information including a weather forecast and events

## Quick FACTS

- The displays are controlled by a single Pi on the rear
- It's housed in an IKEA Ribba picture frame
- The display can be controlled via a webpage
- Custom messages can be entered
- The larger display can also show yellow

# SUBSCRIBE TODAY FROM ONLY £5

## SAVE UP TO 35%

WIN! 10 NEW CLUSTER HATS
The MagPi
The official Raspberry Pi magazine
Issue 77 | January 2019 | magpi.cc
20 Amazing Accessories
Best Raspberry Pi kit & components
1970s Computing
Go back in time with the PiDP-11
MAKE WITH CODE
Learn Python & control electronics with Raspberry Pi
Build a Smart Door
Control your locks with Raspberry Pi
Plus!
Save files in the Command Line
The Raspberry Pi SelfieBot
Gaming with Picade Console
The 10 best DAC sound cards

### Subscriber Benefits

- **FREE Delivery**
  Get it fast and for FREE
- **Exclusive Offers**
  Great gifts, offers, and discounts
- **Great Savings**
  Save up to 35% compared to stores

### Rolling Monthly Subscription

- **Low Monthly Cost** (from £5)
- **Cancel at any time**
- **Free delivery to your door**
- **Available worldwide**

### Subscribe for 12 Months

| | |
|---|---|
| £55 (UK) | £90 (USA) |
| £80 (EU) | £95 (Rest of World) |

Free Raspberry Pi 3A+ with 12 Month upfront subscription only (no Raspberry Pi 3A+ with Rolling Monthly Subscription)

## Subscribe online: magpi.cc/subscribe

# LIMITED OFFER!

# FREE!

## Raspberry Pi 3A+

**WITH YOUR 12 MONTH PRINT SUBSCRIPTION**

WORTH £25

This is a limited offer. It replaces our usual offer of a free Pi Zero W.
Offer subject to change or withdrawal at any time.

BUY IN PRINT WORLDWIDE MAGPI.CC/STORE
The MagPi
The official Raspberry Pi magazine
THE RASPBERRY PI SUPERGUIDE
NEW 2019
• Get going guide • Starter projects • Tech jargon busted
The All New 3A+
Code Pac-Man
PLUS
WIN! 10 Signed 3A+
Make Smart Lights
Scratch Resources | Raspberry Pi Boombox | GFX HAT

Buy now: magpi.cc/subscribe

**SUBSCRIBE on app stores**

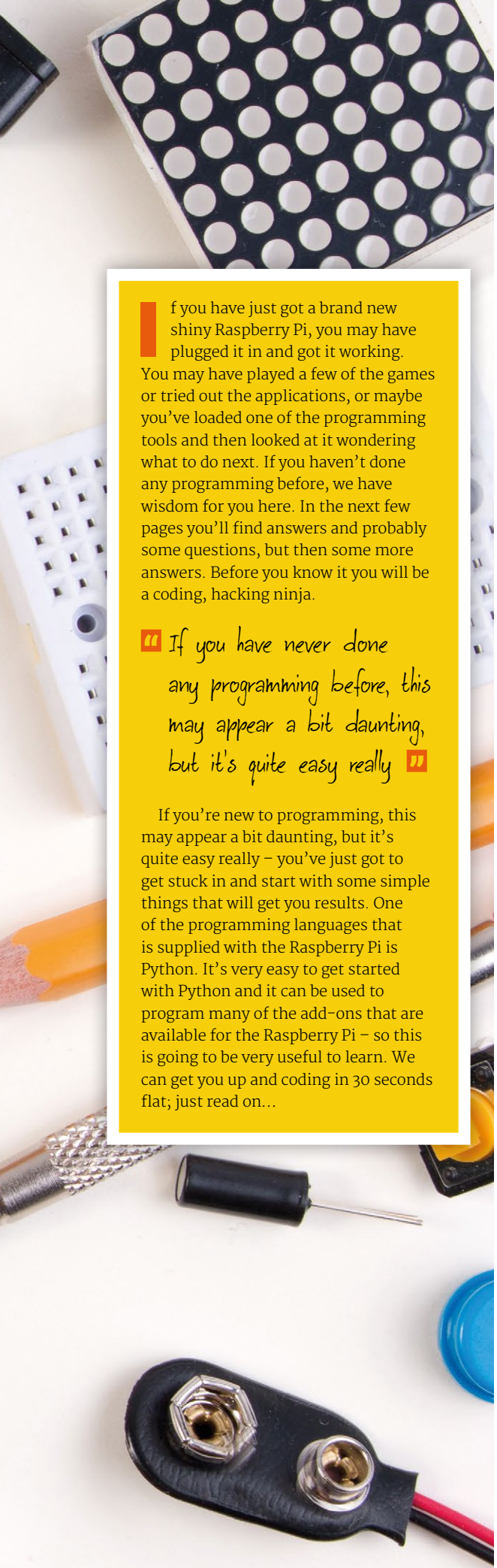
From **£2.29**

Available on the App Store

GET IT ON Google Play

# MAKE WITH CODE

**If you have bought a Raspberry Pi to learn how to code and hack new electronic gizmos, then you have made a great choice. Get started with this guide...**



If you have just got a brand new shiny Raspberry Pi, you may have plugged it in and got it working. You may have played a few of the games or tried out the applications, or maybe you've loaded one of the programming tools and then looked at it wondering what to do next. If you haven't done any programming before, we have wisdom for you here. In the next few pages you'll find answers and probably some questions, but then some more answers. Before you know it you will be a coding, hacking ninja.

> "If you have never done any programming before, this may appear a bit daunting, but it's quite easy really"

If you're new to programming, this may appear a bit daunting, but it's quite easy really – you've just got to get stuck in and start with some simple things that will get you results. One of the programming languages that is supplied with the Raspberry Pi is Python. It's very easy to get started with Python and it can be used to program many of the add-ons that are available for the Raspberry Pi – so this is going to be very useful to learn. We can get you up and coding in 30 seconds flat; just read on...

# MAKE AND RUN A PROGRAM

Mark Vanstone

Educational software author from the nineties, author of the ArcVenture series, disappeared into the corporate software wasteland. Rescued by the Raspberry Pi!

magpi.cc/YiZnxL
@mindexplorers

To get started with coding is really easy. Coding is just giving the Raspberry Pi an instruction to do something. The only thing you need to know is what language to talk to it in. In this case we are going to talk in Python 3. We will need to write our instructions somewhere so, to start, open a Terminal window – click the icon in the top left of the screen which is a grey box with a blue bar across the top. This opens a black window with a prompt: **pi@raspberrypi:~ $** in green. If you type `python3` and hit **ENTER**, Python starts and you will see the triple chevron prompt: **>>>**. Now type `print("Hello")` and hit **ENTER**. Bosh! Your first program. You have instructed your Raspberry Pi to print the word 'Hello' and, all being well, it has obeyed.

Entering programs like this is not going to be very useful most of the time, so now let's look at an app we can write and save a program with. Go to the desktop menu (click the Raspberry Pi logo in the top left of the screen) and in the Programming section, select Thonny (which should then open by default in its Simple mode). Try typing in the following program in the Thonny editor and save it, then run it by clicking the Run button. The output will be displayed in the Shell frame below the program editor.

## mwc1.py

Language: Python
magpi.cc/HNJhhd

```
001. import random
002.
003. correct = False
004. r = random.randint(1,10)
005. c = 0
006. while correct == False:
007.     n = input("Guess my number between 1 and 10: ")
008.     c = c + 1
009.     if int(n) == r:
010.         correct = True
011.     else:
012.         if int(n) > r:
013.             print("Sorry, my number is lower. Try again.")
014.         else:
015.             print("Sorry, my number is higher. Try again")
016.
017. print("Well done. The correct answer was " + str(r) + ".
     You got it in " + str(c) + " tries.")
```

# LEARN TO CODE

Coding involves using several elements. Let's take a look at a range of them and how they work

▲ After initialising the data the program needs, we loop round, updating the answer variable until the player guesses all the letters or they run out of tries

In the last program, we get input from the keyboard and output text and numbers by using the `print()` function. We have also used a condition structure in the form of `if` and `else`. Python is very particular about how you indent the code with spaces (four per indent level); this shows that the indented code is inside another structure. In the last program, everything indented after the `while` statement will be part of that loop. Let's take a look at a few more coding techniques.

## 01 Using lists

We are going to write a Hangman-style game, where we start with a secret word and the player has to guess it, letter by letter. If correct, we show them where that letter appears in the word. They are allowed ten wrong attempts before the game ends. See the **mwc2.py** code to follow along. First, we make a list of words to choose from. A list is defined in Python using square brackets, like: `list = ["a","b","c"]`. We'll call our list `WORDLIST`. In this case we're writing the list name in upper case to show that it is a constant, i.e. It is not going to change throughout the program.

## 02 Pick a word

We can pick a word from our list using the `random` module. We import the module at the top of our code, then we can use the `random.choice()` function to get a word and store it in a variable: `theWord`. When we call a function that is inside a module, we use a full stop between the module name and the function name. Next, we want to get the player to start guessing what the word is. If we look at the bottom of the code, we can see that we call a function called `startGuessing()`. This is our own function that we need to define.

## 03 Defining functions

Each time we call a function, the code inside it runs. Sometimes functions have outputs, like our function `updateAnswer()` that returns the variable `result`. One of the rules of Python is that you must define a function before you call it, so we will need to define our `startGuessing()` function near the top of the code. To do so, we write `def` and then the name of the function, followed by brackets and a colon. If we want to pass variables as inputs into a function, we can add them inside the brackets.

## 04 Getting loopy

Now for the code in our `startGuessing()` function. We set the number of tries and dashes, one for each letter of the secret word, then we go into a loop. The code says: "While the player still has some tries left and the answer we have is not the secret word, run the following code." In our loop, we print the answer we have so far and how

## Coding guides

Want to learn more about coding? Check out our guides to Python coding and object-orientated programming in *The MagPi* #53 and #54 respectively.

**magpi.cc/53**
**magpi.cc/54**

# mwc2.py

Language: Python
magpi.cc/RqQdhR

```
001. import random
002.
003. WORDLIST = ["orange", "table", "january", "balloon",
     "mouse", "speaker", "lorry"]
004. theWord = random.choice(WORDLIST)
005.
006. def startGuessing():
007.     triesLeft = 10
008.     answer = "-" * len(theWord)
009.
010.     while triesLeft > -1 and not answer == theWord:
011.         print("\n" + answer)
012.         print(str(triesLeft) + " tries left")
013.         guess = input("Guess a letter:")
014.         if len(guess) != 1:
015.             print("Just guess one letter at a time.")
016.         elif guess in theWord:
017.             print("Yes that letter is in the word.")
018.             answer = updateAnswer(theWord, answer, guess)
019.         else:
020.             print("Sorry, that letter is not in the word.")
021.             triesLeft -= 1
022.
023.     if triesLeft < 0:
024.         print("Sorry, you have run out of tries. The word was: " + theWord)
025.     else:
026.         print("Well done, You guessed right. The word was: " + theWord)
027.
028. def updateAnswer(word, ans, guess):
029.     result = ""
030.     for i in range(len(word)):
031.         if word[i] == guess:
032.             result = result + guess
033.         else:
034.             result = result + ans[i]
035.     return result
036.
037. print("I'm thinking of a word....")
038. startGuessing()
```

**Module**
A module is another code file. It can contain functions, variables, and data.

**List**
A list is a collection of variables, in this case string variables.

**Loop**
A loop is a part of the code that is repeated one or more times, usually depending on an equation being True.

**Input**
An input is data that a program receives, in this case from the keyboard.

**Variable**
A variable is a container for data, in this case a string which can change as the program runs.

**Function**
A function contains code that can be called from elsewhere in the program.

**Output**
Output is anything that the program produces, in this case words printed to the shell window.

**Condition**
A condition branches a program to execute one set of code or another. Sometimes there are several branches.

**Calling a function**
When a function is called, the code inside the function is run and then returns to the next line of code after the function call.

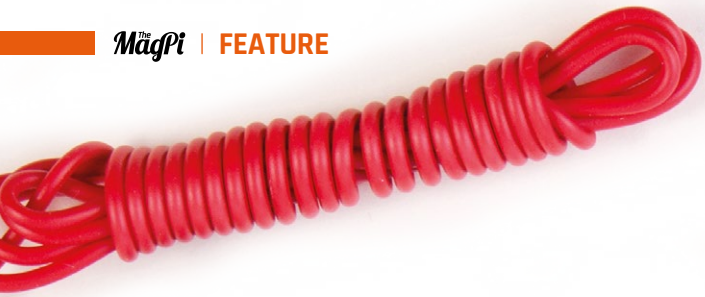
many tries the player has left. Then we use an `if`, `elif`, `else` condition structure to respond to the player, depending on what they typed.

## 05 Changing the answer

If the player guesses a letter correctly, we call another function: `updateAnswer()`. This uses a `for` loop to add the correct letters into our answer variable, then return that string (a variable containing letters/characters rather than a numeric value). This then becomes the answer variable that we print at the start of each loop in the `startGuessing()` function. When the player has guessed all the right letters in our word or they have got it wrong ten times, the program will drop out of the loop to reach the final part of the function.

## 06 Win or lose

We have an `if` and `else` structure to print congratulations, or let the player know they've run out of tries. A few of the functions are used with variables: `len()` finds the length of a string, and `str()` converts a number variable into a string so it can be added to the start or the end of another string. After the function is complete, it returns to where it was called, which is the end of the program.



# CONTROL THINGS WITH CODE

*Now we've got the hang of the coding, let's put it to work with some electronic components*

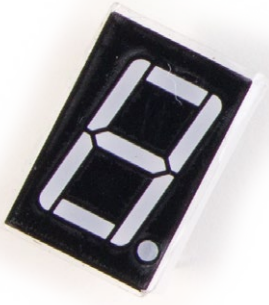
## You'll Need

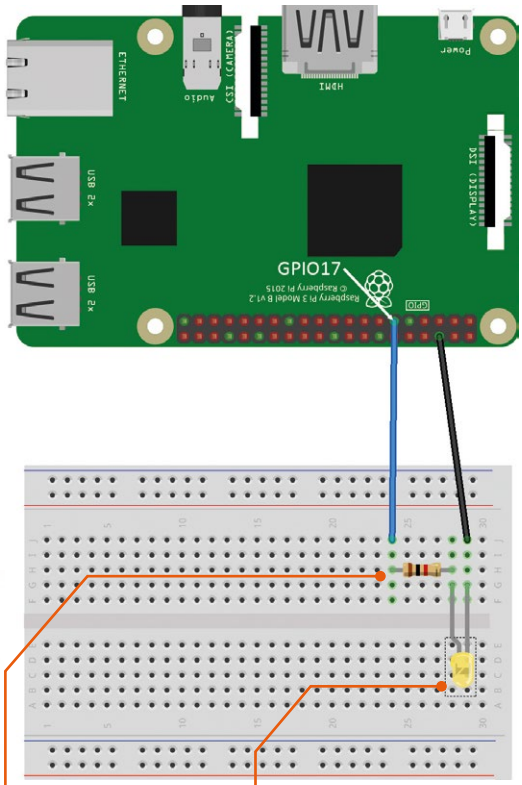
A breadboard
**magpi.cc/NtjSiy**

An LED
**magpi.cc/WBVPxG**

A resistor
**magpi.cc/iDTFag**

2 × male-to-female jumper wires
**magpi.cc/OkMyVX**

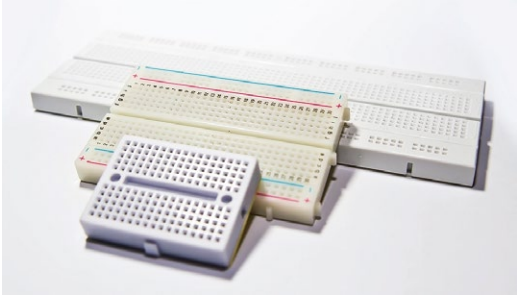



GPIO17
This resistor is 1kΩ (kilohm), but you can use a similar value
This is a yellow LED, but you can buy LEDs in many different colours

In the previous example, we imported a module into our code and used functions from it. If we want to control electronics from code, there is a very useful module available called `gpiozero`. GPIO stands for 'general-purpose input/output' and the line of double pins on one side of the Raspberry Pi are called GPIO pins. For details about the labels of all the pins, see **pinout.xyz**. If we connect electrical components to these GPIO pins, we can use the `gpiozero` module to make things happen. When we import a module, there are often many different functions inside. We can also make new coding 'objects' with them. Objects are like variables but have their own set of functions and properties inside them that we can call or change, and we do that using the same dot notation (full stop) that we did with the `random` module. For more details on using coding objects (called object-oriented programming or OOP), see issue 54 of *The MagPi*: **magpi.cc/54**.

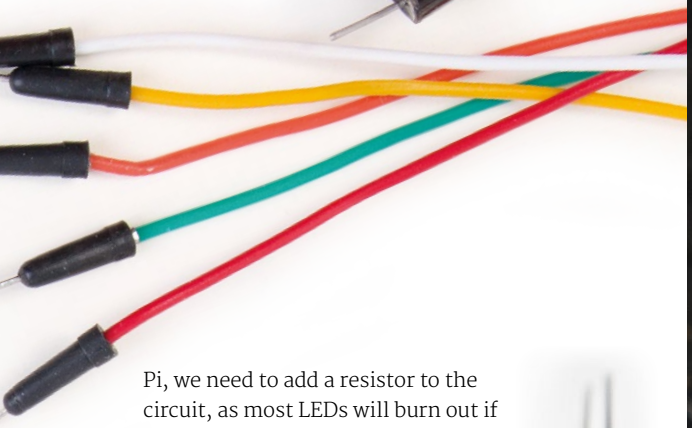
## 01 The breadboard

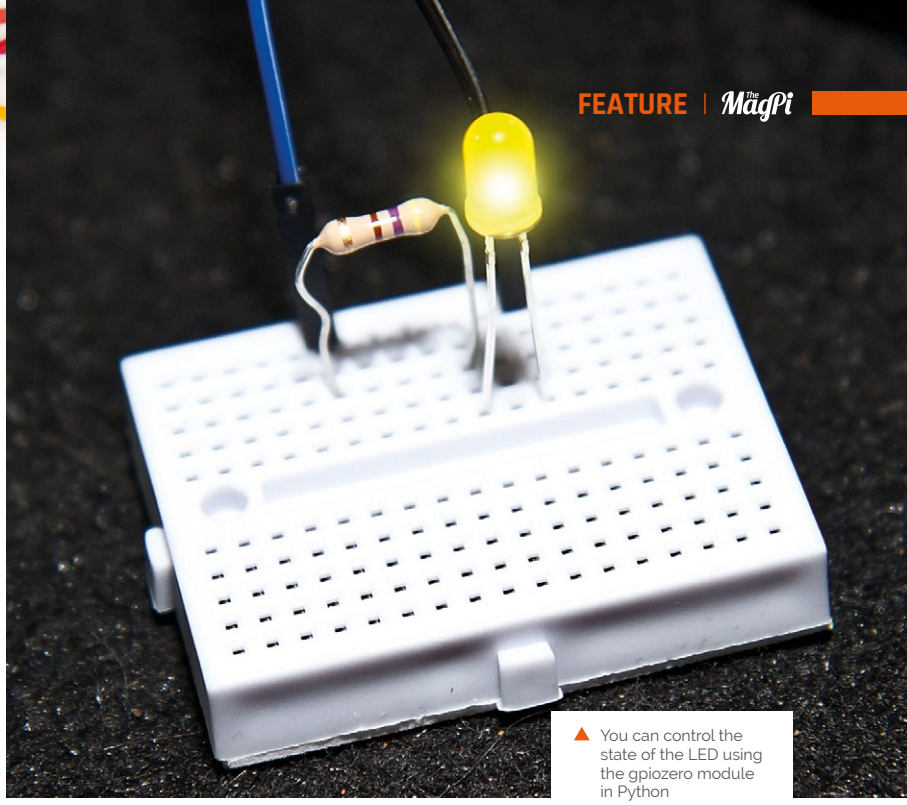
Breadboards come in various sizes, but they all work in the same way. If there are tracks marked red and black/blue (and/or + and −) on the long edges of the board, these are for power and are connected along the length – although sometimes divided into sections. The matrix of holes which make up the main part of the board are connected in lines the other way (vertically in the diagram). There is usually a break in the centre of the board so that the two sides are not connected.



## 02 Light-emitting diode (LED)

An LED is a bit like a bulb in that if you apply electricity to it in the right way, it lights up. An LED is also a diode, which means that the electricity needs to flow in the correct direction or it will not light. When connecting an LED to the Raspberry





▲ You can control the state of the LED using the gpiozero module in Python

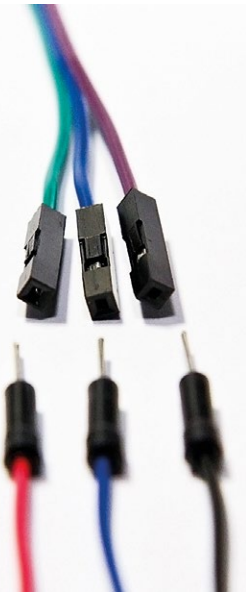
Pi, we need to add a resistor to the circuit, as most LEDs will burn out if we connect them directly to the main power output. In the example, we are using a 1kΩ resistor, but it's fine to use another similar value. To light an LED using `gpiozero`, assemble the components as in the diagram, then write a Python program: start with `from gpiozero import LED`, then create an LED object with `led = LED(17)`, and finally type `led.on()` to light the LED.

## 03 A resistor

Resistors do what their name suggests: they resist the flow of electricity (current). Some components need to have a certain amount of current in order to operate correctly. Resistors enable us to set the current or voltage to a suitable level for the other components we are using. There are many different types of resistors, with different resistance values. The resistance value can be read from the pattern of coloured stripes on the resistor. You can also get variable resistors, which are known as potentiometers.

## 04 Jumper wires

We need to connect our components to the Raspberry Pi GPIO pins. For this we use jumper wires. The ones we will be using have a female connector at one end, to go on the GPIO pins, and a male connector to go in the holes of the breadboard. You can also get jumper wires with both male connectors or both female connectors, for different situations. They can be bought in strips all joined together, sometimes known as 'jumper jerky'.

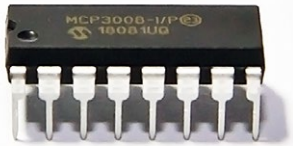


## 05 Potentiometer

A potentiometer is a variable resistor. It usually has a turning spindle that changes the resistance from one value to another, quite often from no resistance to full resistance (no electricity flowing). A potentiometer can provide us with a variable output voltage which we can measure with the GPIO, but there is a slight problem. The potentiometer provides an 'analogue' output (varies continuously between values) and the GPIO inputs are only digital, i.e. on or off. So we need another component: an analogue-to-digital converter (ADC).

## 06 Analogue-to-digital converter

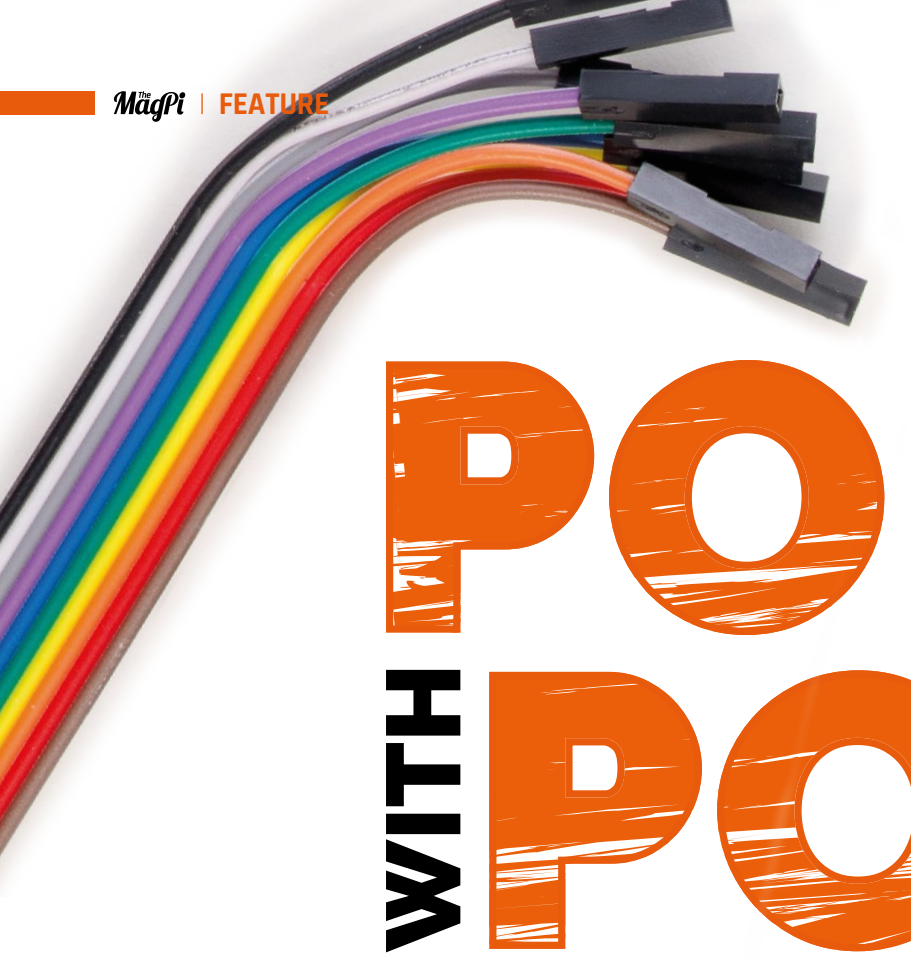
ADC components come in various forms, but the one we have in this example is called an MCP3008. It's an integrated circuit (IC), meaning that it's a box with some circuitry inside it. We don't need to know what is inside it – we just need to know what to connect to each of the legs of the IC. We will need to wire up several of the legs to GPIO pins and provide the IC with power; when we've done that, we can connect the potentiometer to the IC and then read values in showing the position of the potentiometer spindle using the `gpiozero` module. We'll cover the code in the next section.




## Electronics guide

For more details about using electronic components with the Raspberry Pi, check out our Electronics Starter Guide in issue 64 of *The MagPi*.

**magpi.cc/64**



# PONG WITH POTS

Now to put what we have covered to the test: we will make a retro game and control it with our own homemade controllers

## You'll Need

Breadboard
magpi.cc/vhZkGh

6 × male-to-female jumper wires &
10 × male-to-male jumper wires
magpi.cc/ffZNwL

2 × potentiometers
magpi.cc/oZRFEe

MCP3008 integrated circuit
magpi.cc/aCvXuo

On the previous page we talked about potentiometers and analogue-to-digital converters, and this is where we get to use them. It's a bit more complicated than lighting up an LED, but only a little. First, we are going to write a program which has two rectangular bats at each side of the screen that can be moved up and down by two players. A ball bounces backwards and forwards between the bats until one player misses the ball and the other player scores a point. That's right, you guessed it, the game is Pong and we are going to create a controller for each player from a potentiometer and wire it into the Raspberry Pi.

## 01 Super-fast game coding

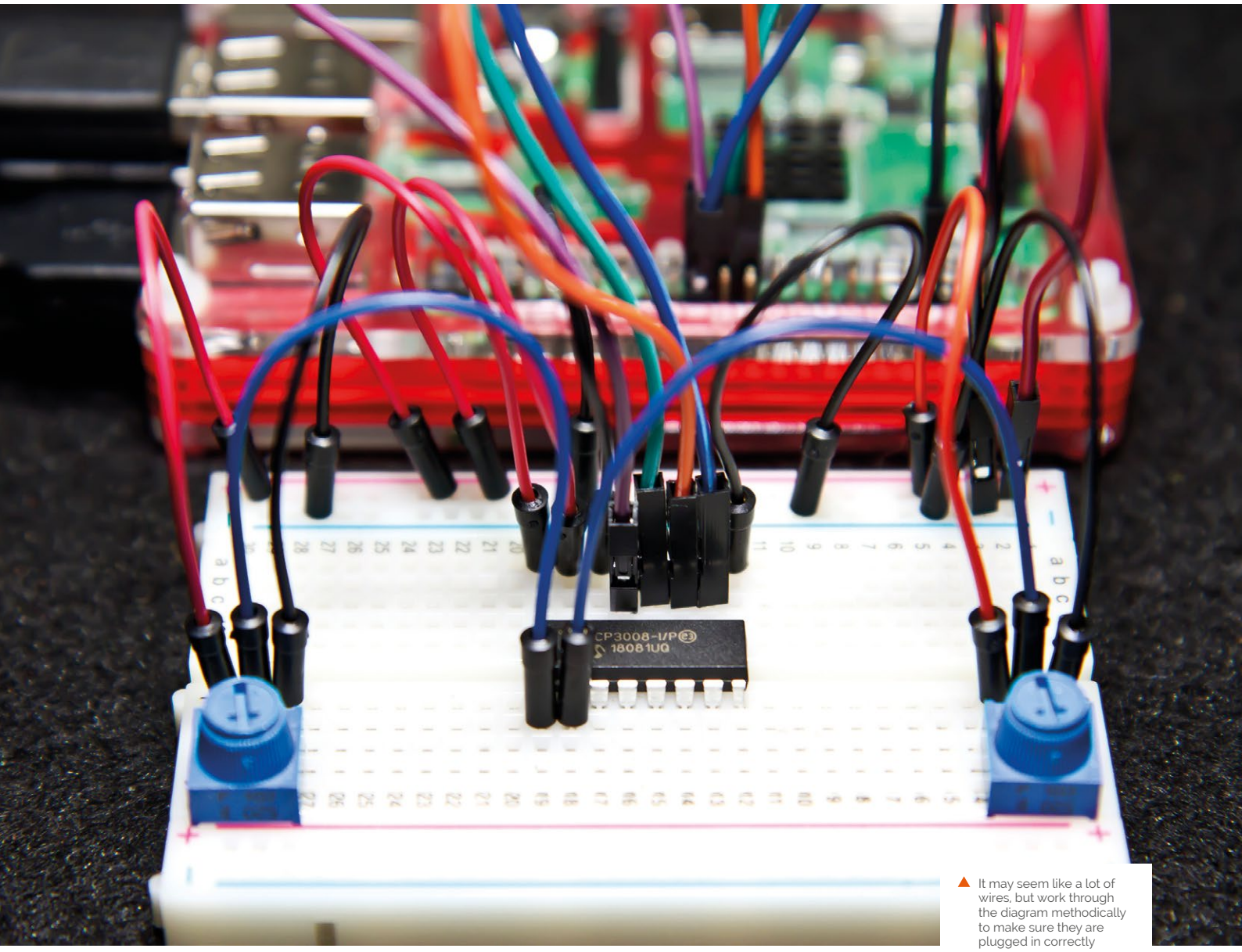
If you have been following other coding articles in recent issues of *The MagPi*, you will know that when writing a quick game on the Raspberry Pi, Pygame Zero is your friend. We can make the basics of the game code very quickly by importing the `pgzrun` module, which holds all the Pygame Zero code. We need to call `pgzrun.go()` at the end of our code, and that's our game window sorted. As with all Pygame Zero programs, we have a `draw()` function to write the graphics to the window, and an `update()` function to update the game items between draw cycles.

## 02 Running the code

The listing **mwc3.py** provides all the code you need for the game to work. There is some code in the `updatePaddles()` function for the keyboard to control the paddles or bats, just in case you want to test it before making the proper controllers. We import several modules with this code. We have covered `pgzrun`, but we also need `random` so that the ball will move in a random direction each time it starts. In addition, we need `gpiozero` to deal with the input from the controllers, and we need the `math` module for calculating the direction of the ball.

## 03 Wiring it all up

One thing to bear in mind when connecting any electronics to a computer is that if the wires are connected in the wrong way, you may cause damage to the computer or the electronic




▲ It may seem like a lot of wires, but work through the diagram methodically to make sure they are plugged in correctly

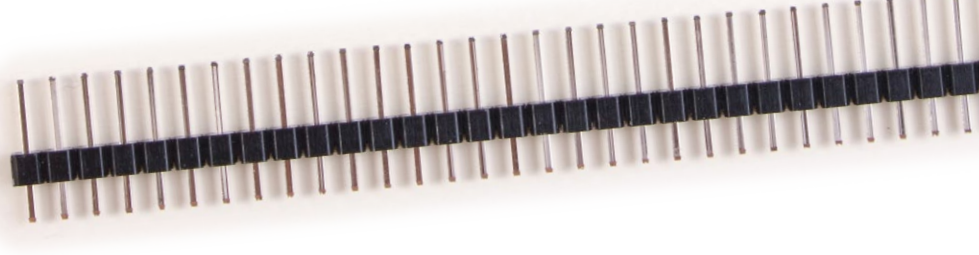
components, so it's always a good idea to power off your Raspberry Pi before connecting anything to the GPIO pins. Follow the wiring diagram (overleaf) carefully, making sure that the jumper wires are connected to the right GPIO pins and to the correct places on the breadboard. Once you have put everything in place, it's a good idea to have another check just to make sure.

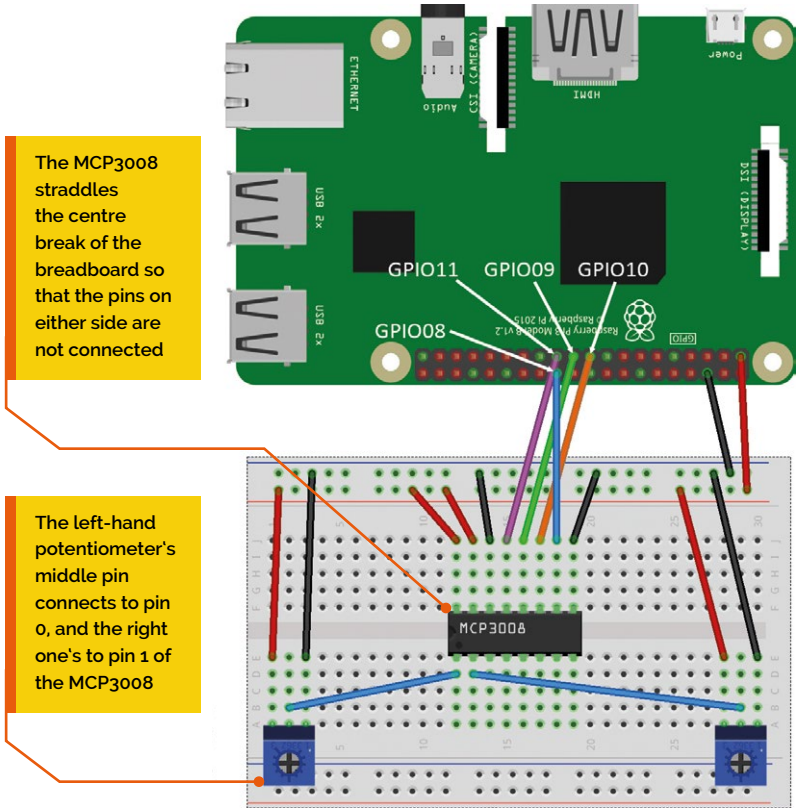
## 04 The MCP3008 IC

The MCP3008 converts the voltage from our potentiometers into a number with the help of the `gpiozero` module. It has eight channels for input, but we are going to just use two of them in this case. You will see from the diagram that all the legs on the top side of the IC are connected to either GPIO pins or to power lines. There are two red connections going to the positive power track, then a black lead to the negative or ground track. Then there are four coloured wires that go to: purple – GPIO 11; green – GPIO 09; orange – GPIO 10; and blue – GPIO 08. Then there is one last connector to the ground track.

## 05 The inputs

All the MCP3008 pins on the bottom side of the IC are input channels. We will use the first two pins, which are channel 0 and channel 1. We





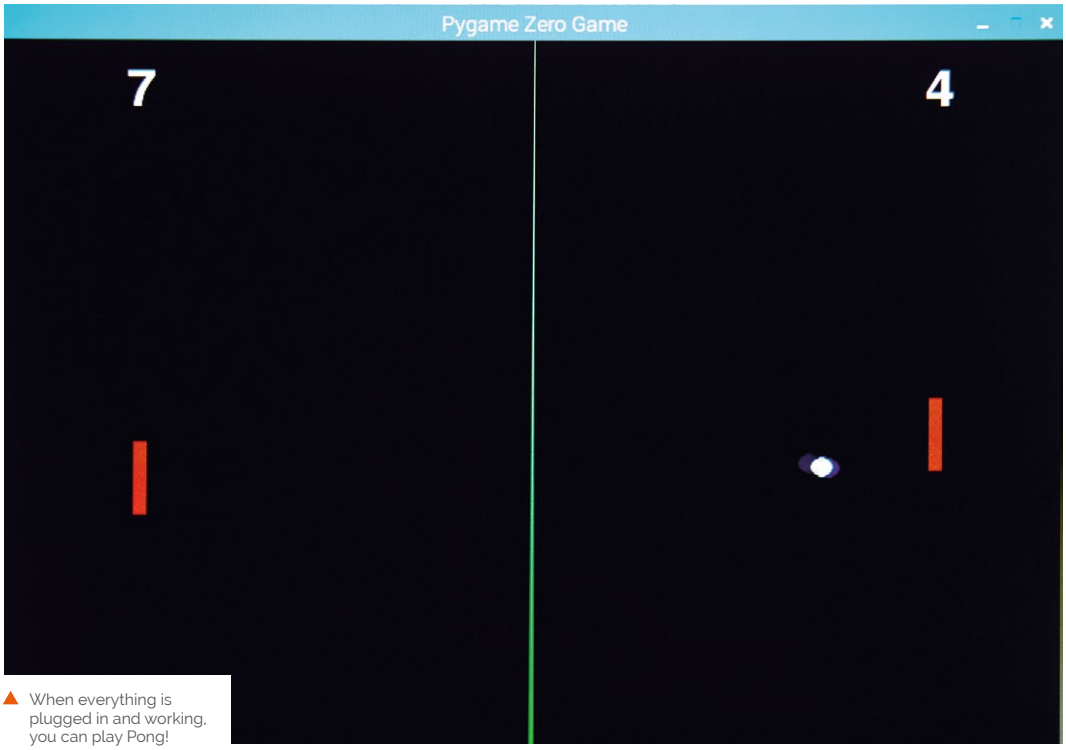

The MCP3008 straddles the centre break of the breadboard so that the pins on either side are not connected

The left-hand potentiometer's middle pin connects to pin 0, and the right one's to pin 1 of the MCP3008

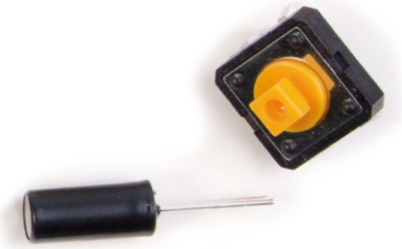
connect the middle pins of our potentiometers to those channel pins, which will read the positions and convert that to a value in our program. If you want to know exactly what all the pins are for on this IC, you can do a web search for 'MCP3008 pinout' and that will give you descriptions of each. ICs are very useful in electronics as they mean that we can reduce how complex our circuits are and we don't need to know exactly how they work inside. They are a little bit like modules in Python.

## 06 Finishing the job

When you have checked that all the connections are correct, you can switch the Raspberry Pi back on and reload your program. You may want to initialise the SPI interface on your Raspberry Pi by going to the main desktop menu > Preferences > Raspberry Pi Configuration, and go to the Interfaces tab. It will work without this, but may cause a few warnings in the Thonny shell window. So, all being well, when you run your program you will have a game of Pong which can be controlled by two players with the potentiometers. If it doesn't work first time, check your code and then your wiring, and try again.




▲ When everything is plugged in and working, you can play Pong!



# mwc3.py

Language: Python
magpi.cc/umUcfq

```
import pgzrun
import random
from gpiozero import MCP3008
import math

pot1 = MCP3008(0)
pot2 = MCP3008(1)

# Set up the colours
BLACK = (0  ,0  ,0  )
WHITE = (255,255,255)
p1Score = p2Score = 0
BALLSPEED = 5
p1Y = 300
p2Y = 300

def draw():
    screen.fill(BLACK)
    screen.draw.line((400,0),(400,600),"green")
    drawPaddles()
    drawBall()
    screen.draw.text(str(p1Score) , center=(105, 40), color=WHITE, fontsize=60)
    screen.draw.text(str(p2Score) , center=(705, 40), color=WHITE, fontsize=60)

def update():
    updatePaddles()
    updateBall()

def init():
    global ballX, ballY, ballDirX, ballDirY
    ballX = 400
    ballY = 300
    a = random.randint(10, 350)
    while (a > 80 and a < 100) or (a > 260 and a < 280):
        a = random.randint(10, 350)
    ballDirX = math.cos(math.radians(a))
    ballDirY = math.sin(math.radians(a))

def drawPaddles():
    global p1Y, p2Y
    p1rect = Rect((100, p1Y-30), (10, 60))
    p2rect = Rect((700, p2Y-30), (10, 60))
    screen.draw.filled_rect(p1rect, "red")
    screen.draw.filled_rect(p2rect, "red")

def updatePaddles():
    global p1Y, p2Y

    p1Y = (pot1.value * 540) +30

    p2Y = (pot2.value * 540) +30

    if keyboard.up:
        if p2Y > 30:
            p2Y -= 2
    if keyboard.down:
        if p2Y < 570:
            p2Y += 2
    if keyboard.w:
        if p1Y > 30:
            p1Y -= 2
    if keyboard.s:
        if p1Y < 570:
            p1Y += 2

def updateBall():
    global ballX, ballY, ballDirX, ballDirY, p1Score, p2Score
    ballX += ballDirX*BALLSPEED
    ballY += ballDirY*BALLSPEED
    ballRect = Rect((ballX-4,ballY-4),(8,8))
    p1rect = Rect((100, p1Y-30), (10, 60))
    p2rect = Rect((700, p2Y-30), (10, 60))
    if checkCollide(ballRect, p1rect) or checkCollide(ballRect, p2rect):
        ballDirX *= -1
    if ballY < 4 or ballY > 596:
        ballDirY *= -1
    if ballX < 0:
        p2Score += 1
        init()
    if ballX > 800:
        p1Score += 1
        init()


def checkCollide(r1,r2):
    return (
        r1.x < r2.x + r2.w and
        r1.y < r2.y + r2.h and
        r1.x + r1.w > r2.x and
        r1.y + r1.h > r2.y
    )

def drawBall():
    screen.draw.filled_circle((ballX, ballY), 8, "white")
    pass

init()
pgzrun.go()
```

Part 01

# Pi Bakery: In a spin

Enhance your Raspberry Pi by adding a VGA output and a retro 8-bit sound chip emulator using the eight-core Propeller chip

MAKER

**Mike Cook**

Veteran magazine author from the old days, writer of the Body Build series, plus co-author of *Raspberry Pi for Dummies*, *Raspberry Pi Projects*, and *Raspberry Pi Projects for Dummies*.

**magpi.cc/kLYEZs**

Ever wanted an extra display on your Raspberry Pi, maybe for a two-player game or to log extra information? Then dust off that old VGA monitor and put it to good use to make a text and graphics display. Ever wanted to recreate those authentic 8-bit sounds of yesteryear without trying to buy long obsolete chips? Well now you can, easily and cheaply, by using the remarkable but much neglected Propeller chip. We will show you how to do it, as well as making a Propeller development board for even more projects.

## 01 The spin processor

The Propeller spin processor has been around for some time now and, in many ways, it was ahead of its time. It has eight processors that are called 'cogs'; they all operate in parallel and can run at speeds in excess of 96MHz. Each one has a turn at controlling the output pins, hence the name 'spin'. This is summarised in **Figure 1**. In addition to the cogs, there is memory: both RAM and ROM, which are shared resources

## You'll Need

- P8X32A processor **magpi.cc/pNdgpi**
- 24LC245 EEPROM **magpi.cc/ctpjtt**
- 5MHz crystal **magpi.cc/inNNCq**
- USB to TTL serial adapter 3.3V 5.5V **magpi.cc/ViLqJr**
- Full parts list **magpi.cc/pptCFz**

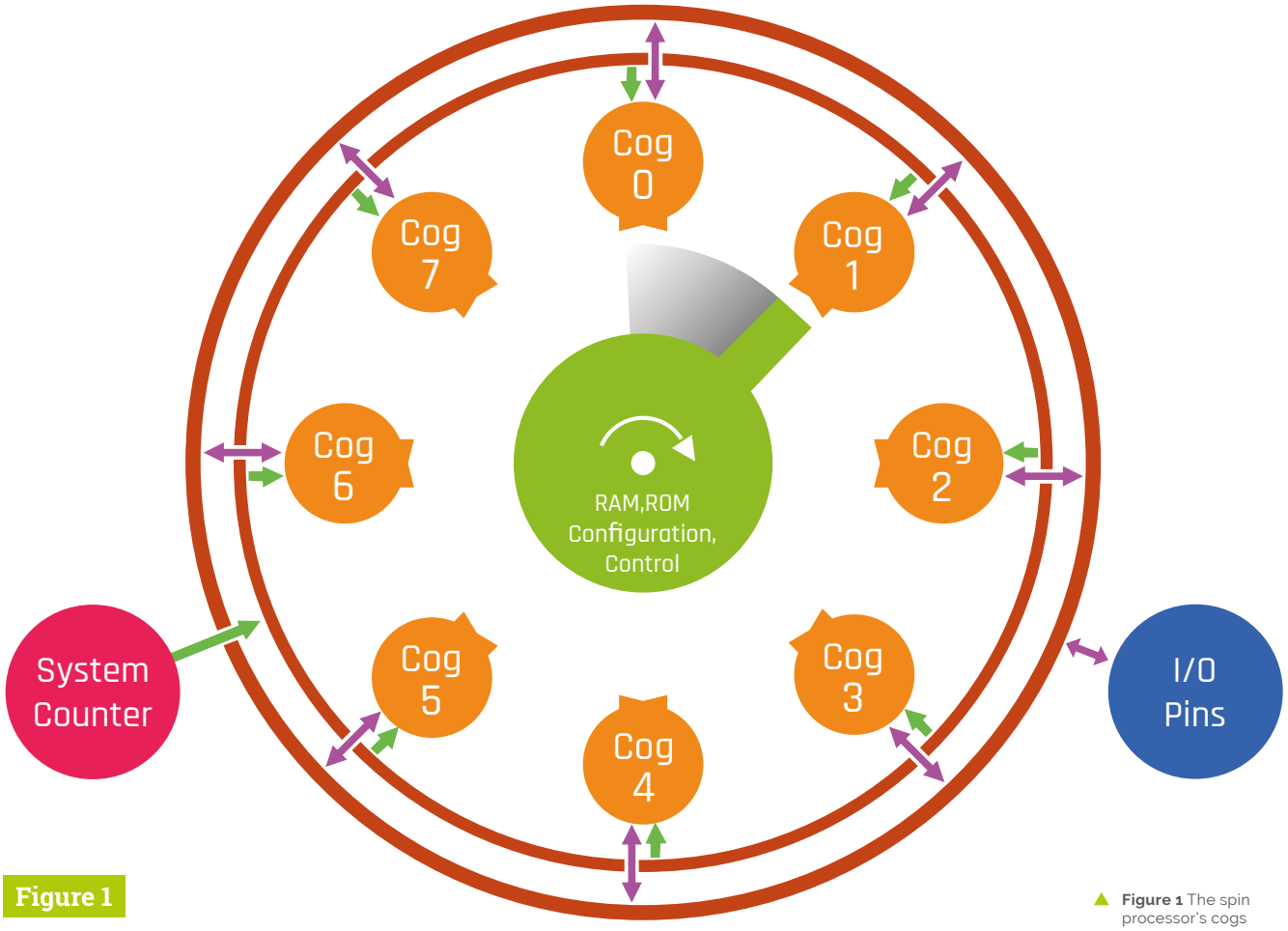



Figure 1

▲ **Figure 1** The spin processor's cogs

▼ **Figure 2** The block diagram of our system

for all the cogs. On bootup, code can be loaded automatically from an external EEPROM, so in effect the processor can run any code without specifically loading it.

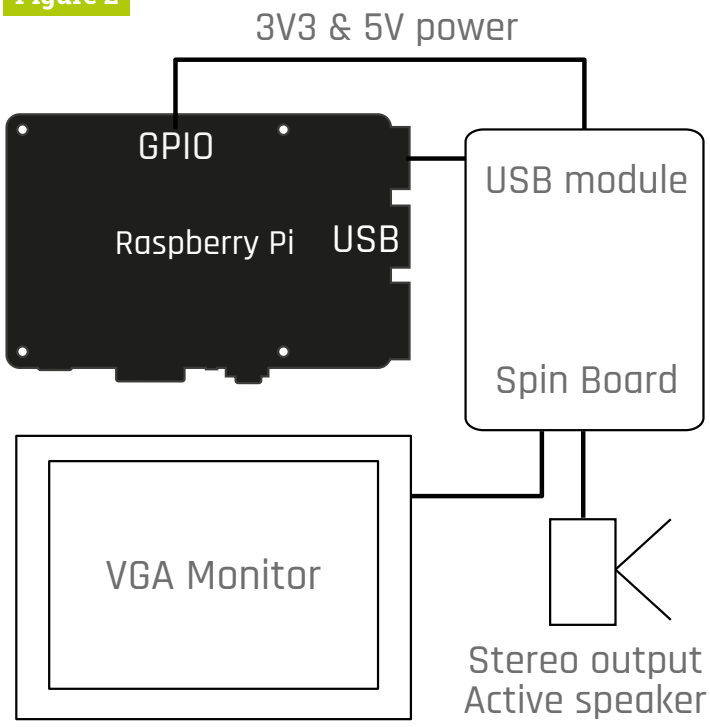
## 02 Spin language

Built into the processor is an interpreter for a language called Spin, which has a simple syntax and is quite easy to learn. You can also run machine code, for those tasks that require it. In a similar way to the Arduino, preprogrammed cog codes can easily be dropped in, just like libraries, and there is a cog exchange website where you can download over a hundred cogs. In effect, you can think of cogs as programmable peripherals, and that is what we want to do with them here.

## 03 Our project

What we are going to do is to build a development board for this processor and make a VDU display, and a retro sound chip emulator with it, and show you how to drive these new

Figure 2



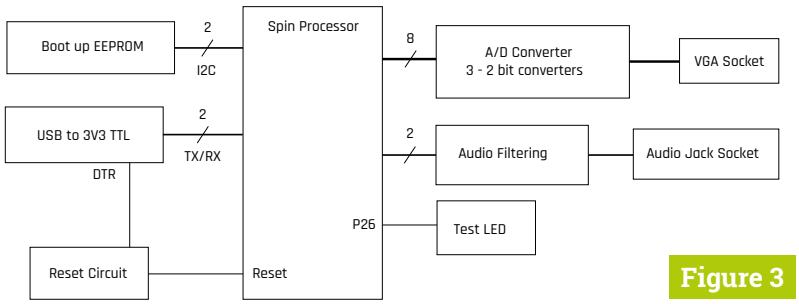



Figure 3

▲ **Figure 3** The block diagram of our spin board

peripherals with Python. **Figure 2** (previous page) shows the block diagram of our project. With the introduction of the Raspberry Pi 3, there has been a change to the serial port on the GPIO pins; so to allow our project to seamlessly work with any model, we have chosen the USB route to access it, but taking its power from the Pi itself.

## 04 Our board

**Figure 3** shows the block diagram of the board. A block diagram is useful for getting the overall structure of what we are going to make, without delving into all the detail we need to add to make a schematic. Connections between blocks that consist of more than one wire are often shown, as here, by a thick line with a slash and number above it, to indicate the number of wires involved. As well as the video and audio outputs, we have included an LED to flash, and there are 15 unused I/O pins on the processor for further expansion.

## 05 The full schematic

**Figure 4** shows the full schematic of the board. By reference to the block diagram, you can identify the main functional units. The three video components – R, G, and B – are made using two pins, each joined together through resistors, to form a simple DAC (digital-to-analogue converter). For best results, these resistors should

▼ **Figure 4** The schematic of our spin board

Figure 4

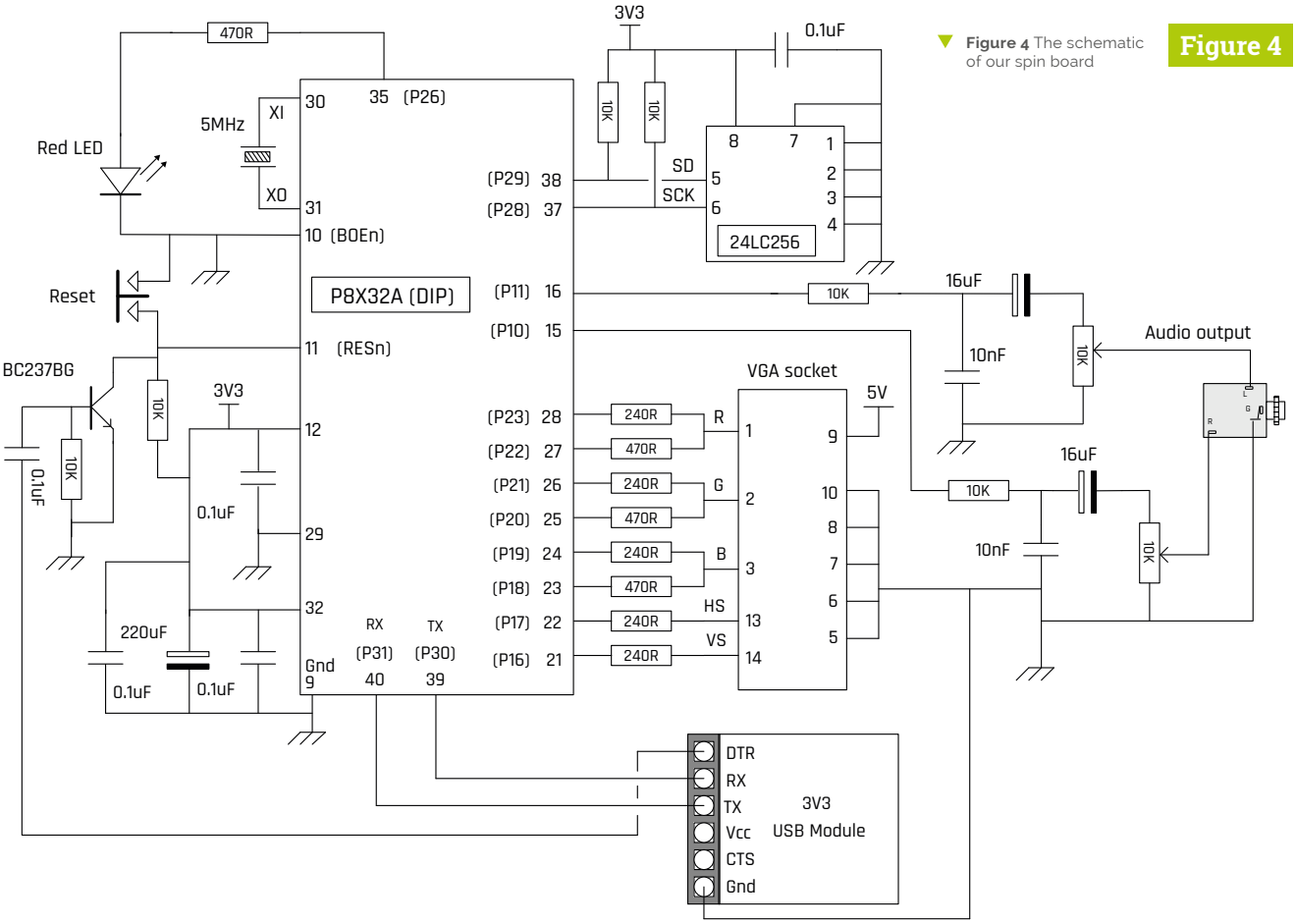
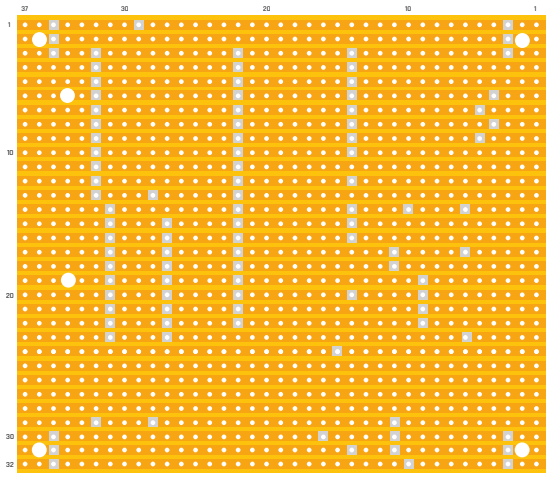


Figure 5



be accurate to 1%. The audio circuits consist of a 10 kΩ resistor and 10 nF capacitor to form a restoration filter, which will remove a lot of the PWM noise. Then a capacitor provides AC coupling, and a final trim pot allows you to adjust the output.

## 06 Preparing the board

The project is constructed on a piece of 32-strip, 37-hole stripboard – see **Figure 5**. The

◀ **Figure 5** The track side of our spin board

▼ **Figure 6** The physical layout of our spin board

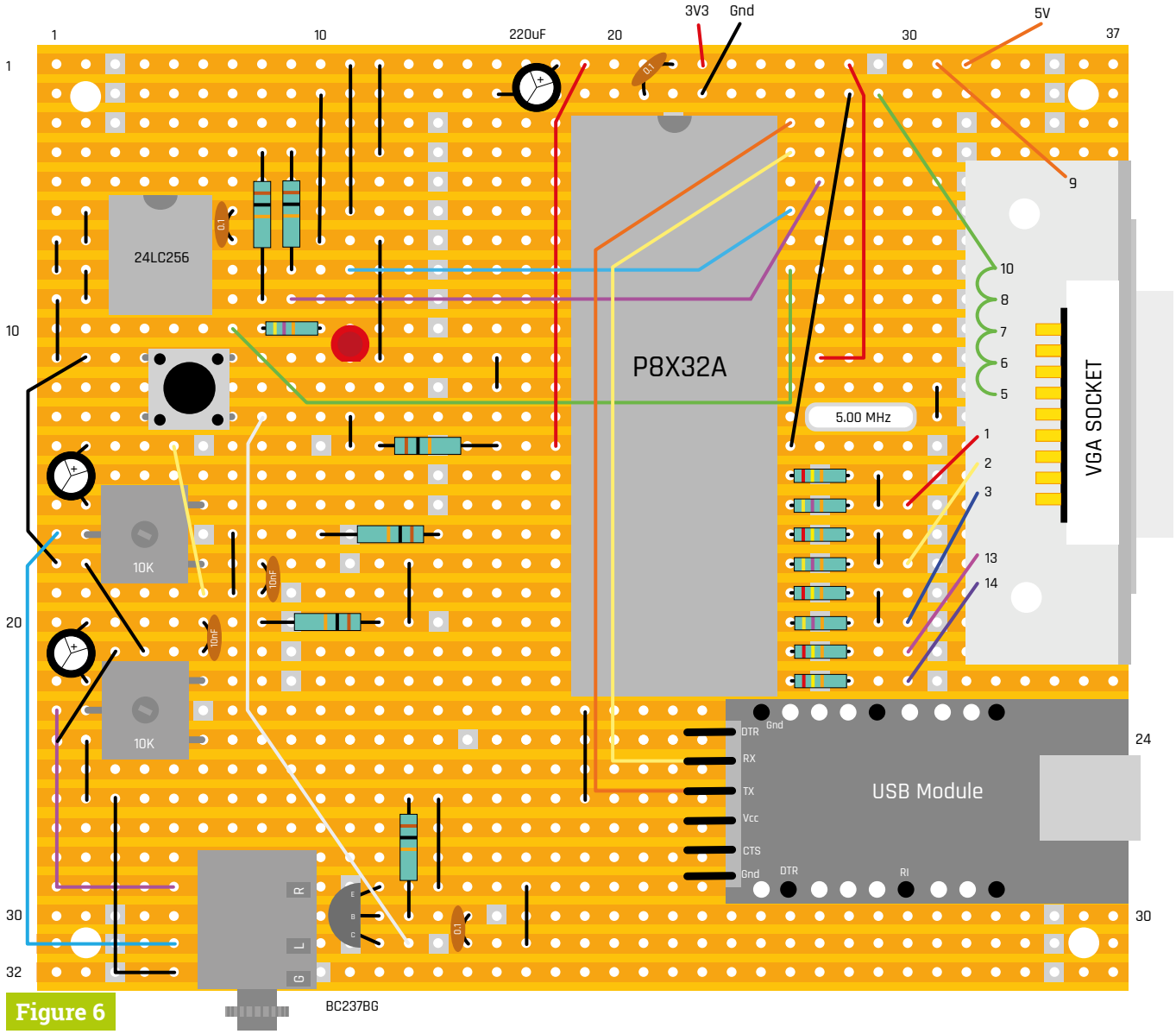
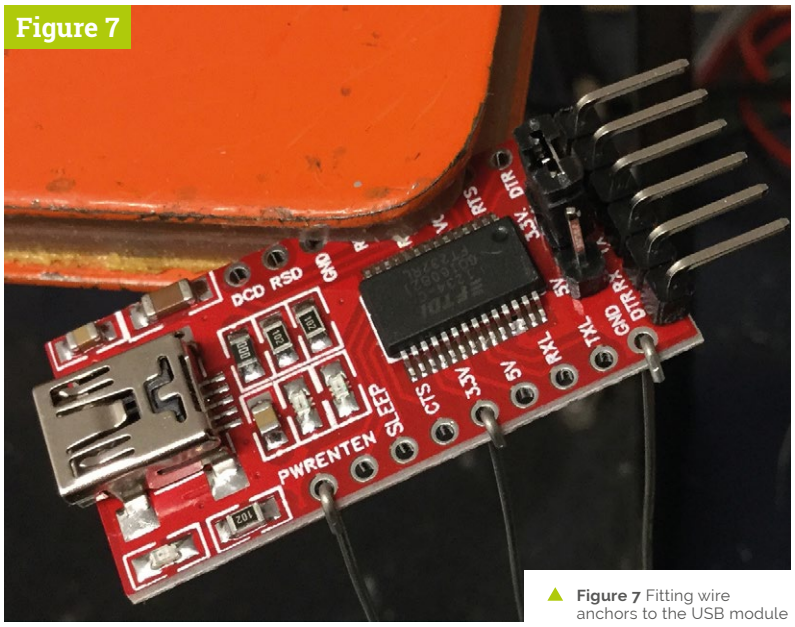



Figure 6

Figure 7




▲ **Figure 7** Fitting wire anchors to the USB module

grey squares indicate where the track has been cut around a hole. Each hole has a coordinate, given by a strip and hole number; strip numbers go from top to bottom and, on the underside, hole numbers go from right to left. Drill four 3 mm board mounting holes at S2 H36 (Strip 2 Hole 36), S2 H2, S31 H2, and S31 H36. Add the VGA socket mounting at S6 H34 and S19 H34, and fit the audio jack. Make all the cuts in the tracks.

## 07 Building the board

**Figure 6** (previous page) shows the components – on this side, hole numbers go from left to right. The 'hidden detail' of the track breaks and the strip outlines can be seen through the board. Start by soldering the 40-pin socket with pin 1 at S3 H19. Then add a 0.1 µF capacitor at S11 H21 and S11 H23, and another at S14 H21 and S14 H23. As these will be inside the socket, push them flat against the board so they don't foul the chip. Next, add the 8-pin socket with pin 1 at S6 H3.

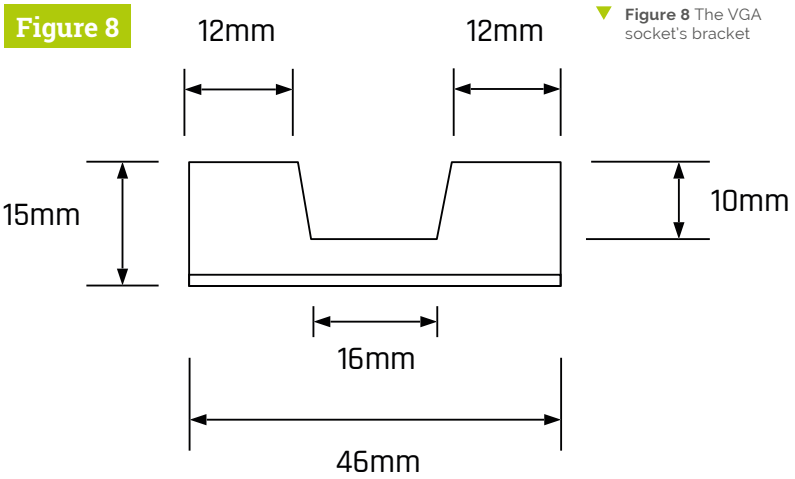
## 08 Adding the components

Now, add all the wire links; these are the vertical black lines. Then add all the resistors and the push-button. Next, add the LED – note the flat on the lower side, indicating the LED's cathode or negative end. Add the capacitors, pots, crystal, and transistor. The USB module is soldered to the board (**Figure 7**) using six thick wires in the positions indicated by the black holes in the diagram. Only two are used as signals – the others are for mechanical support only. Make sure you make the link on the module so that you get 3V3 (3.3 V) signals out of it.

## 09 The VGA socket

The socket is mounted on a bracket made from a 46 mm length of 15×15 mm angle aluminium (**Figure 8**). This requires a trapezoidal slot to be cut out of it using a saw and file. The saw cuts you could make are shown in **Figure 9**; finish off by filing so the socket slots in neatly. Slot in the socket and mark its mounting holes

Figure 8




▼ **Figure 8** The VGA socket's bracket

▼ **Figure 9** Saw cuts to remove the majority of the metal

Figure 9

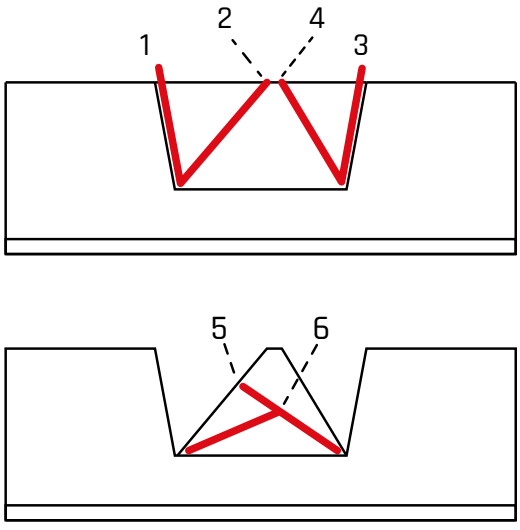
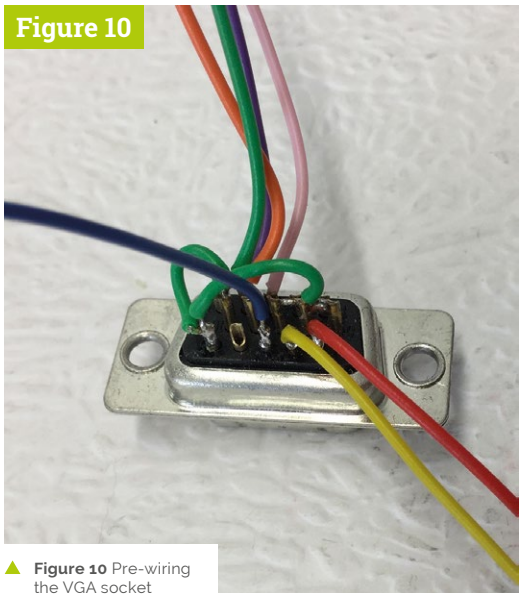
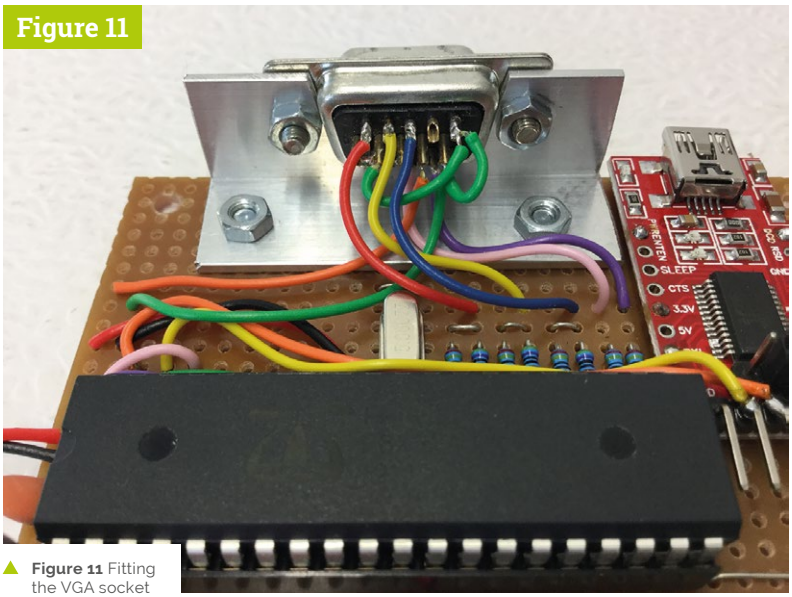


Figure 10



▲ **Figure 10** Pre-wiring the VGA socket

Figure 11



▲ **Figure 11** Fitting the VGA socket

through the socket's holes. Do the same with the bracket's mounting holes on the board. Drill out with a 3 mm drill and fix the bracket to the board.

## 10 Wiring the VGA socket

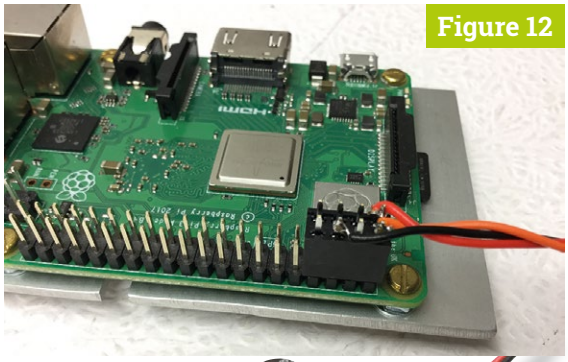
It is best if you add wires into the socket (**Figure 10**) before mounting it on the board. Add 40 mm wires to pin numbers 1, 2, 3, 9, 13, and 14. Use different colours of wires for each one so they can easily identified once mounted. For the pin numbers 5, 6, 7, 8, and 10, you have to chain these together. Do this by stripping the ends off two wires and twisting them together, and then tinning them – that is, adding a bit of solder. Then trim off the exposed wire to about 4 mm and insert in the hole of the pin, then solder it up.

## 11 Adding the connection wires

Screw the VGA socket to the bracket and wire up the flying leads to the points shown (**Figure 11**), trimming them to be only the length they need to be. Make the rest of the wires linking up the circuit. The 3V3, 5 V, and ground on the top of the layout go to the Raspberry Pi. We used three wires of 300 mm connected to an 8-pin header socket pushed into the GPIO pins (**Figure 12**). Some VGA monitors don't require the 5 V signal, so that could be omitted if that is the case with yours.

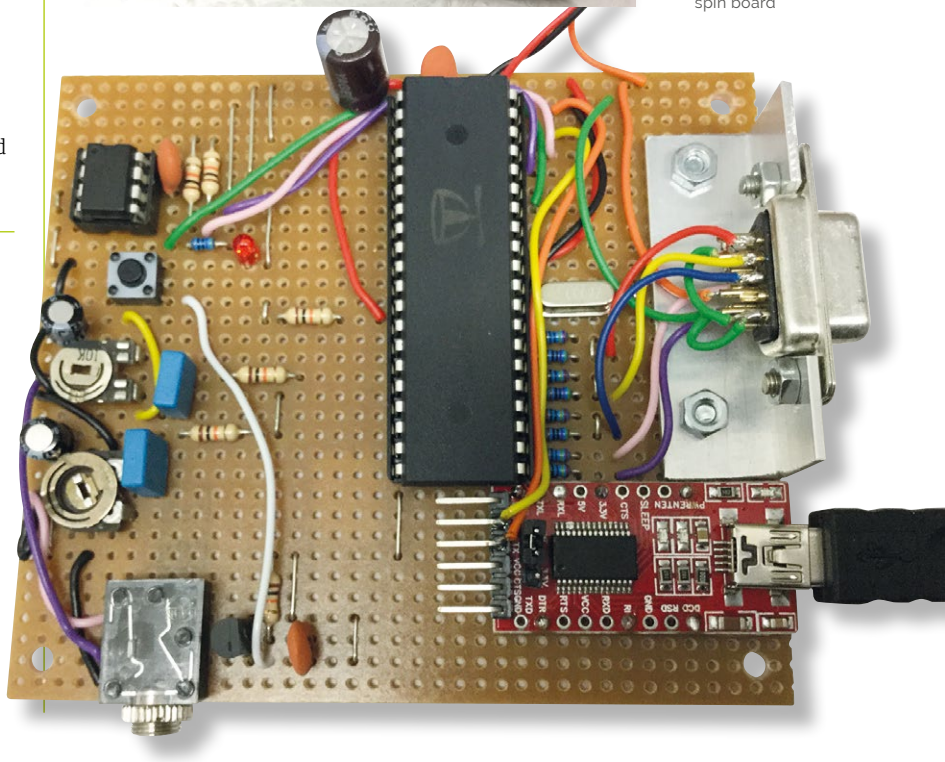
With the hardware built, next month we will see how to program our spin board, test it, and get some funky sounds from it.

Figure 12



◀ **Figure 12** Tapping the power from the Pi

▼ The finished spin board



Part 07

# Code your own Pac-Man game: part 2

Pac-Man captured the hearts and pocket money of many young people in the eighties. In part two of our guide, we add some groovy features to the basic game created last month

MAKER

**Mark Vanstone**

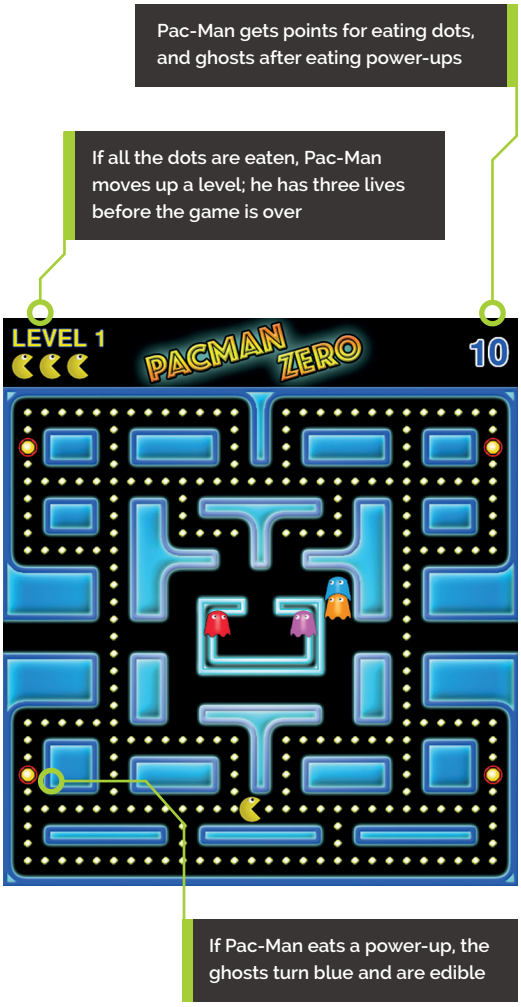
Educational software author from the nineties, author of the ArcVenture series, disappeared into the corporate software wasteland. Rescued by the Raspberry Pi!

magpi.cc/YiZnxl

@mindexplorers

## You'll Need

- Raspbian Jessie or newer
- An image manipulation program such as GIMP, or images available from **magpi.cc/nBSXKz**
- The latest version of Pygame Zero (1.2)
- USB joystick or gamepad (optional)
- Headphones or speakers


Pac-Man gets points for eating dots, and ghosts after eating power-ups
If all the dots are eaten, Pac-Man moves up a level; he has three lives before the game is over
LEVEL 1
PACMAN ZERO
10
If Pac-Man eats a power-up, the ghosts turn blue and are edible

In part one, we created a maze for our player to move around, and restricted movement to just the corridors. We provided some dots to eat and some ghosts to avoid. In this part we are going to give the ghosts some more brains so that they are a bit more challenging to the player. We will also add the bonus power-ups which turn the ghosts into tasty edibles, give Pac-Man some extra levels to explore and some extra lives. So far in this series we have not dealt with music, so we will have a go at putting some music and sound effects into the game.

## 01 Need more brains

In part one, we left our ghosts wandering around the maze randomly without much thought for what they were doing, which was a bit unfair as Pac-Man could evade them without too much trouble. In the original game, each ghost had a program that it followed to characterise its movements. We are going to add some brains to two of the ghosts. The first we will make follow Pac-Man, and the second we will get to ambush by moving ahead of Pac-Man. We will still leave in some random movement, otherwise it may get a bit too difficult.

## 02 Follow the leader

First, let's get the red ghost to follow Pac-Man. We already have a `moveGhosts()` function

Adding a brain to a ghost to follow the player

from part one and we can add a condition to see if we are dealing with the first ghost: `if g == 0: followPlayer(g, dirs)`. This calls `followPlayer()` if it's the first ghost. The `followPlayer()` function receives a list of directions that the ghost can move in. It then tests the x coordinate of the player against the x coordinate of the ghost and, if the direction is valid, sets the ghost direction to move toward the player. Then it does the same with the y coordinates.

## 03 Y over x

The keen-witted among you will have noticed that if x and y movements towards the player are both valid, then the y direction will always win. We could throw in another random number to choose between the two, but in testing this arrangement it doesn't cause any significant problem with the movement. See **figure1.py** for the `followPlayer()` function. You will see there is a special condition `aboveCentre()` when we check the downward movement. We are checking that the ghost is not just above the centre, otherwise it will go back into its starting enclosure.

## 04 The central problem

If we go back to the `moveGhosts()` function, we need another centre-related condition: `if inTheCentre(ghosts[g])`. This is because if we leave the ghost to randomly move around our centre enclosure, it may take a long time to get out. In part one, you may have noticed that from time to time one ghost would get stuck in the centre. What we do is, if we detect that a ghost is in the centre, we always default to direction 3, which is up. If we run the game with this condition and the `followPlayer()` function, we should see all the ghosts making their way straight out of the centre and then the red ghost making a bee-line towards Pac-Man.

## figure1.py

```
001. def followPlayer(g, dirs):
002.     d = ghosts[g].dir
003.     if d == 1 or d == 3:
004.         if player.x > ghosts[g].x and dirs[0] == 1:
005.             ghosts[g].dir = 0
006.         if player.x < ghosts[g].x and dirs[2] == 1:
007.             ghosts[g].dir = 2
008.     if d == 0 or d == 2:
009.         if player.y > ghosts[g].y and dirs[1] == 1 and not
    aboveCentre(ghosts[g]): ghosts[g].dir = 1
010.         if player.y < ghosts[g].y and dirs[3] == 1:
011.             ghosts[g].dir = 3
012.
013.
014. def aboveCentre(ga):
015.     if ga.x > 220 and ga.x < 380 and ga.y > 300 and ga.y
    < 320:
016.         return True
017.     return False
```

## figure2.py

```
001. # This code goes in the update() function
002.
003.     if player.status == 1:
004.         i = gameinput.checkInput(player)
005.         if i == 1:
006.             player.status = 0
007.             player.x = 290
008.             player.y = 570
009.
010. # This code goes in the gameinput module
011. # in the checkInput() function
012.
013.     if joystick_count > 0:
014.         jb = joyin.get_button(1)
015.     else:
016.         jb = 0
017.     if p.status == 1:
018.         if key.get_pressed()[K_RETURN] or jb:
019.             return 1
```

### 05 It's an ambush!

So, the next brain to implant is for the second ghost. We will add a function `ambushPlayer()` in the same way we did for the first ghost, but this time `if g == 1:`. The `ambushPlayer()` function works very much like the `followPlayer()` function, but this time we just check the direction that Pac-Man is currently moving and try to move in that direction. We, of course, cannot know which direction the player is going to move, and this may seem a bit of a simplistic approach to ambushing the player, but it is surprising how many times Pac-Man ends up wedged between these two ghosts with this method.

### 06 Scores on the doors

Brain functions could be added to all the ghosts, but we are going to leave the ghost brains for now as there is plenty more to do to get our

▲ Checking to see if **ENTER** or button A has been pressed and resetting the player actor

▼ Pac-Man gets three lives. You can use the gamepad or joystick buttons to ask for input from the player

game completed. Before we go any further, we ought to get a scoring system going and reward Pac-Man for all the dots eaten. We can attach the score variable to the player actor near the top of our code with `player.score = 0` and then each time a dot is eaten we add 10 to the score with `player.score += 10`. We can also display the score in the `draw()` function (probably top right is best) with `screen.draw.text()`.

## 07 Three strikes and you're out!

As is the tradition in arcade games, you get three lives before it's game over. If you followed our previous tutorial for Space Invaders, you will know how we do this. We just add a lives variable to the player actor and then each time Pac-Man is caught by a ghost, we take a life off, set `player.status = 1`, and print a message to say press **ENTER**. When pressed, we set `player.status = 0` and send Pac-Man back to the starting place. Then we continue. Have a look at **figure2.py** to see the code we add to reset Pac-Man to the start.

## 08 Printing lives

We have the system for keeping track of the `player.lives` variable, but we also need to show the player how many lives they have left. We can do this with a simple loop like we used in the previous Space Invaders tutorial. We can have a `drawLives()` function which we call from our `draw()` function. In that function, we go round a loop for the number of lives we have by saying `for l in range(player.lives):` and then we can use the same image that we use for the player and say `screen.blit("pacman_o", (10+(l*32),40))`.

## 09 Which button to press

You may notice in **figure2.py** that in our gameinput module we are checking a joystick button as well as the **ENTER** key. You may want to do a few tests with the gamepads or joysticks that you're using, as the buttons may have different numbers. You can also prompt the player to press (in this case) the A button to continue. If you were designing a game that relied on several buttons being used, you might want to set up a way of mapping the buttons to values depending on what type of gamepad or joystick is being used.

## figure3.py

```
001. # This code is in our main code file (pacman2.py)
002.
003. def initDots():
004.     global pacDots
005.     pacDots = []
006.     a = x = 0
007.     while x < 30:
008.         y = 0
009.         while y < 29:
010.             d = gamemaps.checkDotPoint(10+x*20, 10+y*20)
011.             if d == 1:
012.                 pacDots.append(Actor("dot",(10+x*20,
    90+y*20)))
013.                 pacDots[a].status = 0
014.                 pacDots[a].type = 1
015.                 a += 1
016.             if d == 2:
017.                 pacDots.append(Actor("power",(10+x*20,
    90+y*20)))
018.                 pacDots[a].status = 0
019.                 pacDots[a].type = 2
020.                 a += 1
021.             y += 1
022.         x += 1
023.
024. # This code is in the gamemaps module
025.
026. def checkDotPoint(x,y):
027.     global dotimage
028.     if dotimage.get_at((int(x), int(y))) ==
    Color('black'):
029.         return 1
030.     if dotimage.get_at((int(x), int(y))) == Color('red'):
031.         return 2
032.     return False
```

▲ Updated code to include the creation of power-ups

## 10 I have the power!

The next item on our list is power-ups. These are large glowing dots that, when eaten, turn all the ghosts dark blue. In their blue form they can be eaten for bonus points and they return to the centre of the maze. First, let's devise a way to place the power-ups in the maze. We have updated the **pacmandotmap.png** image to include some red squares, instead of black, in the positions where we want our power-ups to be. Then, when we initialise our dots and call `checkDotPoint(x,y)`, we look for red as well as black – **figure3.py** shows how we change our code to do this.

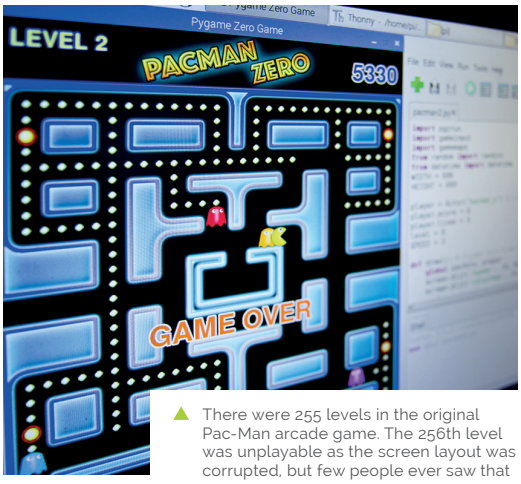
## gamemaps.py

> Language: **Python 3**

```
001. from pygame import image, surface, Color
002. moveimage = image.load('images/pacmanmovemap.png')
003. dotimage = image.load('images/pacmandotmap.png')
004.
005. def checkMovePoint(p):
006.     global moveimage
007.     if p.x+p.movex < 0: p.x = p.x+600
008.     if p.x+p.movex > 600: p.x = p.x-600
009.     if moveimage.get_at((int(p.x+p.movex), int(p.y+
     p.movey-80))) != Color('black'):
010.         p.movex = p.movey = 0
011.
012. def checkDotPoint(x,y):
013.     global dotimage
014.     if dotimage.get_at((int(x), int(y))) ==
     Color('black'):
015.         return 1
016.     if dotimage.get_at((int(x), int(y))) ==
     Color('red'):
017.         return 2
018.     return False
019.
020. def getPossibleDirection(g):
021.     global moveimage
022.     if g.x-20 < 0:
023.         g.x = g.x+600
024.     if g.x+20 > 600:
025.         g.x = g.x-600
026.     directions = [0,0,0,0]
027.     if g.x+20 < 600:
028.         if moveimage.get_at((int(g.x+20),
     int(g.y-80))) == Color('black'): directions[0] = 1
029.     if g.x < 600 and g.x >= 0:
030.         if moveimage.get_at((int(g.x), int(g.y-60)))
     == Color('black'): directions[1] = 1
031.     if g.x-20 >= 0:
032.         if moveimage.get_at((int(g.x-20),
     int(g.y-80))) == Color('black'): directions[2] = 1
033.     if g.x < 600 and g.x >= 0:
034.         if moveimage.get_at((int(g.x), int(g.y-100)))
     == Color('black'): directions[3] = 1
035.     return directions
```

### 11 Not all dots are the same

We now have a system to place our power-ups in the maze. The next thing to do is to change what happens when Pac-Man eats a power-up compared to a normal dot. At the moment we just add ten points to the player's score if a dot is eaten, so we need to add more code to handle the event of a power-up being eaten. In the `draw()` function, where we look to see if the player has collided with a dot using `collidepoint()`, we then check the status of the dot (to make sure it's still there) and after this we can add a new condition: `if pacDots[a].type == 2:`.




▲ There were 255 levels in the original Pac-Man arcade game. The 256th level was unplayable as the screen layout was corrupted, but few people ever saw that

### 12 High status ghosts

As we have determined that we are dealing with a power-up (type 2), we can add a loop that goes through the list of ghosts and changes the status of the ghost. Normally the status for a ghost is 0. What we are going to do is change the status to a fairly high number (try 1200 to start with). This will indicate that the ghosts are in their alternate state and we will use the status as a countdown. We will decrement this value each time `update()` is called; when it reaches 0, the ghosts will turn back to normal.

### 13 Why so blue?

To make our ghost turn blue, we are going to add some conditions to our `drawGhosts()` function. We want them to be blue when the status is more than 0, but just to make it interesting we will make them flash when they are about to turn back. So we can write `if ghosts[g].status > 200 or (ghosts[g].status > 1 and ghosts[g].status%2 == 0): ghosts[g].image = "ghost5"`. What this is saying is that if the status is over 200 then make the ghost blue, but if it's less that 200 but greater than 1 then make it blue every other frame. We then have an `else` condition underneath that will set the image to its normal colour.

## gameinput.py

Language: **Python 3**

```
001. from pygame import joystick, key
002. from pygame.locals import *
003.
004. joystick.init()
005. joystick_count = joystick.get_count()
006.
007. if(joystick_count > 0):
008.     joyin = joystick.Joystick(0)
009.     joyin.init()
010.
011. def checkInput(p):
012.     global joyin, joystick_count
013.     xaxis = yaxis = 0
014.     if p.status == 0:
015.         if joystick_count > 0:
016.             xaxis = joyin.get_axis(0)
017.             yaxis = joyin.get_axis(1)
018.         if key.get_pressed()[K_LEFT] or xaxis < -0.8:
019.             p.angle = 180
020.             p.movex = -20
021.         if key.get_pressed()[K_RIGHT] or xaxis > 0.8:
022.             p.angle = 0
023.             p.movex = 20
024.         if key.get_pressed()[K_UP] or yaxis < -0.8:
025.             p.angle = 90
026.             p.movey = -20
027.         if key.get_pressed()[K_DOWN] or yaxis > 0.8:
028.             p.angle = 270
029.             p.movey = 20
030.     if joystick_count > 0:
031.         jb = joyin.get_button(1)
032.     else:
033.         jb = 0
034.     if p.status == 1:
035.         if key.get_pressed()[K_RETURN] or jb:
036.             return 1
037.     if p.status == 2:
038.         if key.get_pressed()[K_RETURN] or jb:
039.             return 1
```

### 14 The tables have turned

Now we have our ghosts all turning blue when a power-up is eaten, we need to change what happens when Pac-Man collides with them. Instead of taking a life from the `player.lives` variable, we are going to add to the `player.score` variable and send the ghost back to the centre. So, the first job is to add a condition in `update()` when we check the ghost `collidepoint()` with the player, which would be `if ghosts[g].status > 0:`. We then add 100 to the `player.score` and `animate()` the ghost back to the centre. See **figure4.py** for the updated code.

### 15 Back to the start

You will notice that when Pac-Man comes into contact with a dark blue ghost, we just animate the actor straight back to the centre in the same time that we normally animate a ghost from one position to the next. This is so that we don't hold up the animation on the other ghosts waiting for the eaten one to get back to the centre. In the original game, the ghosts would turn into a pair of eyes and then make their way back to the centre along the corridors, but that would take too much extra code for this tutorial.

## figure4.py

```
001. # This code is in the update() function
002.
003.         for g in range(len(ghosts)):
004.             if ghosts[g].status > 0: ghosts[g].status -= 1
005.             if ghosts[g].collidepoint((player.x,
     player.y)):
006.                 if ghosts[g].status > 0:
007.                     player.score += 100
008.                     animate(ghosts[g], pos=(290, 370),
     duration=1/SPEED, tween='linear',
     on_finished=flagMoveGhosts)
009.                 else:
010.                     player.lives -= 1
011.                     if player.lives == 0:
012.                         player.status = 3
013.                     else:
014.                         player.status = 1
```

▲ Updated ghost collision code to send them back to the centre if Pac-Man eats them

### 16 Time for some music

So far in this series, we have not covered adding music to games. In the documentation of Pygame Zero, music is labelled as experimental, so we will just have to try it out and see what happens. In the sample GitHub files for this

tutorial, there is a directory called **music** and in that directory is an MP3 file that we can use as eighties arcade game background music. To start our music, all we need to do is write `music.play("pm1")` in our `init()` function to start the **music/pm1.mp3** file. You may also want to set the volume with `music.set_volume(0.3)`.

## 17 More sound effects

The MP3 file will continue playing in a loop until we stop it, so when the game is over (`player.lives = 0`) we can fade the music out with `music.fadeout(3)`. At this stage we can also add some sound effects for when Pac-Man is eating dots. We have a sound in our **sounds** directory called **pac1.mp3** which we will use for this purpose and we can add a line of code just before we animate the player: `sounds.pac1.play()`. This will play the sound every time Pac-Man moves. We can do the same with **pac2.mp3** when a life is lost.

## 18 Level it up

The last thing we need to put into our game is to allow the player to progress to the next level when all the dots have been eaten. We could incorporate several things to make each level harder, but for the moment let's concentrate on resetting the screen and changing the level. If we define our level variable near the top of our code as `level = 0`, then inside our `init()` function we say `level += 1`, then each time we call `init()` we will increase our level variable. This means that instead of saying that the player has won, we just prompt them to continue, and call `init()` to reset everything and level up.

## 19 So much to do

The Pac-Man game has many more things that can be added to it. The original had bonus fruits to collect, the ghosts would move faster as the levels continued, there were animations between some of the levels, and the power-ups would run out quicker. You could add all of these things to this game, but we will have to leave you to do that yourself. Take a look into the history of the Pac-Man game – it's fascinating – and we will be starting a new Pygame Zero game in the next instalment of this series.

## pacman2.py

> Language: **Python 3**

```
001. import pgzrun
002. import gameinput
003. import gamemaps
004. from random import randint
005. from datetime import datetime
006. WIDTH = 600
007. HEIGHT = 660
008.
009. player = Actor("pacman_o") # Load in the player Actor image
010. player.score = 0
011. player.lives = 3
012. level = 0
013. SPEED = 3
014.
015. def draw(): # Pygame Zero draw function
016.     global pacDots, player
017.     screen.blit('header', (0, 0))
018.     screen.blit('colourmap', (0, 80))
019.     pacDotsLeft = 0
020.     for a in range(len(pacDots)):
021.         if pacDots[a].status == 0:
022.             pacDots[a].draw()
023.             pacDotsLeft += 1
024.         if pacDots[a].collidepoint((player.x, player.y)):
025.             if pacDots[a].status == 0:
026.                 if pacDots[a].type == 2:
027.                     for g in range(len(ghosts)): ghosts[g].status = 1200
028.                 else:
029.                     player.score += 10
030.             pacDots[a].status = 1
031.     if pacDotsLeft == 0: player.status = 2
032.     drawGhosts()
033.     getPlayerImage()
034.     player.draw()
035.     drawLives()
036.     screen.draw.text("LEVEL "+str(level) , topleft=(10, 10), owidth=0.5,
    ocolor=(0,0,255), color=(255,255,0) , fontsize=40)
037.     screen.draw.text(str(player.score) , topright=(590, 20), owidth=0.5,
    ocolor=(255,255,255), color=(0,64,255) , fontsize=60)
038.     if player.status == 3: drawCentreText("GAME OVER")
039.     if player.status == 2: drawCentreText(
    "LEVEL CLEARED!\nPress Enter or Button A\nto Continue")
040.     if player.status == 1: drawCentreText(
    "CAUGHT!\nPress Enter or Button A\nto Continue")
041.
042. def drawCentreText(t):
043.     screen.draw.text(t , center=(300, 434), owidth=0.5,
    ocolor=(255,255,255), color=(255,64,0) , fontsize=60)
044.
045. def update(): # Pygame Zero update function
046.     global player, moveGhostsFlag, ghosts
```

DOWNLOAD THE FULL CODE:
magpi.cc/TDtRaV

```
    if player.status == 0:
        if moveGhostsFlag == 4: moveGhosts()
        for g in range(len(ghosts)):
            if ghosts[g].status > 0: ghosts[g].status -= 1
            if ghosts[g].collidepoint((player.x,
player.y)):
                if ghosts[g].status > 0:
                    player.score += 100
                    animate(ghosts[g], pos=(290, 370),
duration=1/SPEED, tween='linear',
on_finished=flagMoveGhosts)
                else:
                    player.lives -= 1
                    sounds.pac2.play()
                    if player.lives == 0:
                        player.status = 3
                        music.fadeout(3)
                    else:
                        player.status = 1
        if player.inputActive:
            gameinput.checkInput(player)
            gamemaps.checkMovePoint(player)
            if player.movex or player.movey:
                inputLock()
                sounds.pac1.play()
                animate(player, pos=(player.x + player.
movex, player.y + player.movey), duration=1/SPEED,
tween='linear', on_finished=inputUnLock)
    if player.status == 1:
        i = gameinput.checkInput(player)
        if i == 1:
            player.status = 0
            player.x = 290
            player.y = 570
    if player.status == 2:
        i = gameinput.checkInput(player)
        if i == 1:
            init()

def init():
    global player, level
    initDots()
    initGhosts()
    player.x = 290
    player.y = 570
    player.status = 0
    inputUnLock()
    level += 1
    music.play("pm1")
    music.set_volume(0.2)

def drawLives():
    for l in range(player.lives): screen.blit("pacman_o",
(10+(l*32),40))

def getPlayerImage():
    global player
    dt = datetime.now()
    a = player.angle
    tc = dt.microsecond%(500000/SPEED)/(100000/SPEED)
    if tc > 2.5 and (player.movex != 0 or player.movey
!=0):
        if a != 180:
            player.image = "pacman_c"
        else:
            player.image = "pacman_cr"
    else:
        if a != 180:
            player.image = "pacman_o"
        else:
            player.image = "pacman_or"
    player.angle = a

def drawGhosts():
    for g in range(len(ghosts)):
        if ghosts[g].x > player.x:
            if ghosts[g].status > 200 or (ghosts[g].status
> 1 and ghosts[g].status%2 == 0):
                ghosts[g].image = "ghost5"
            else:
                ghosts[g].image = "ghost"+str(g+1)+"r"
        else:
            if ghosts[g].status > 200 or (ghosts[g].status
> 1 and ghosts[g].status%2 == 0):
                ghosts[g].image = "ghost5"
            else:
                ghosts[g].image = "ghost"+str(g+1)
        ghosts[g].draw()

def moveGhosts():
    global moveGhostsFlag
    dmoves = [(1,0),(0,1),(-1,0),(0,-1)]
    moveGhostsFlag = 0
    for g in range(len(ghosts)):
        dirs = gamemaps.getPossibleDirection(ghosts[g])
        if inTheCentre(ghosts[g]):
            ghosts[g].dir = 3
        else:
            if g == 0: followPlayer(g, dirs)
            if g == 1: ambushPlayer(g, dirs)

        if dirs[ghosts[g].dir] == 0 or randint(0,50) == 0:
```

```
            d = -1
            while d == -1:
                rd = randint(0,3)
                if aboveCentre(ghosts[g]) and rd == 1:
                    rd = 0
                if dirs[rd] == 1:
                    d = rd
            ghosts[g].dir = d
        animate(ghosts[g], pos=(ghosts[g].x
+ dmoves[ghosts[g].dir][0]*20, ghosts[g].y +
dmoves[ghosts[g].dir][1]*20), duration=1/SPEED,
tween='linear', on_finished=flagMoveGhosts)

def followPlayer(g, dirs):
    d = ghosts[g].dir
    if d == 1 or d == 3:
        if player.x > ghosts[g].x and dirs[0] == 1:
ghosts[g].dir = 0
        if player.x < ghosts[g].x and dirs[2] == 1:
ghosts[g].dir = 2
    if d == 0 or d == 2:
        if player.y > ghosts[g].y and dirs[1] == 1 and not
aboveCentre(ghosts[g]): ghosts[g].dir = 1
        if player.y < ghosts[g].y and dirs[3] == 1:
ghosts[g].dir = 3


def ambushPlayer(g, dirs):
    d = ghosts[g].dir
    if player.movex > 0 and dirs[0] == 1: ghosts[g].dir = 0
    if player.movex < 0 and dirs[2] == 1: ghosts[g].dir = 2

    if player.movey > 0 and dirs[1] == 1 and not
aboveCentre(ghosts[g]): ghosts[g].dir = 1
    if player.movey < 0 and dirs[3] == 1: ghosts[g].dir = 3

def inTheCentre(ga):
    if ga.x > 220 and ga.x < 380 and ga.y > 320 and ga.y <
420:
        return True
    return False

def aboveCentre(ga):
    if ga.x > 220 and ga.x < 380 and ga.y > 300 and ga.y <
320:
        return True
    return False

def flagMoveGhosts():
    global moveGhostsFlag
    moveGhostsFlag += 1

def ghostCollided(ga,gn):
    for g in range(len(ghosts)):
        if ghosts[g].colliderect(ga) and g != gn:
            return True
    return False

def initDots():
    global pacDots
    pacDots = []
    a = x = 0
    while x < 30:
        y = 0
        while y < 29:
            d = gamemaps.checkDotPoint(10+x*20, 10+y*20)
            if d == 1:
                pacDots.append(Actor("dot",(10+x*20,
90+y*20)))
                pacDots[a].status = 0
                pacDots[a].type = 1
                a += 1
            if d == 2:
                pacDots.append(Actor("power",(10+x*20,
90+y*20)))
                pacDots[a].status = 0
                pacDots[a].type = 2
                a += 1
            y += 1
        x += 1

def initGhosts():
    global ghosts, moveGhostsFlag
    moveGhostsFlag = 4
    ghosts = []
    g = 0
    while g < 4:
        ghosts.append(Actor("ghost"+str(g+1),(270+(g*20),
370)))
        ghosts[g].dir = randint(0, 3)
        ghosts[g].status = 0
        g += 1

def inputLock():
    global player
    player.inputActive = False

def inputUnLock():
    global player
    player.movex = player.movey = 0
    player.inputActive = True

init()
pgzrun.go()
```

# THE OFFICIAL Raspberry Pi Beginner's Guide

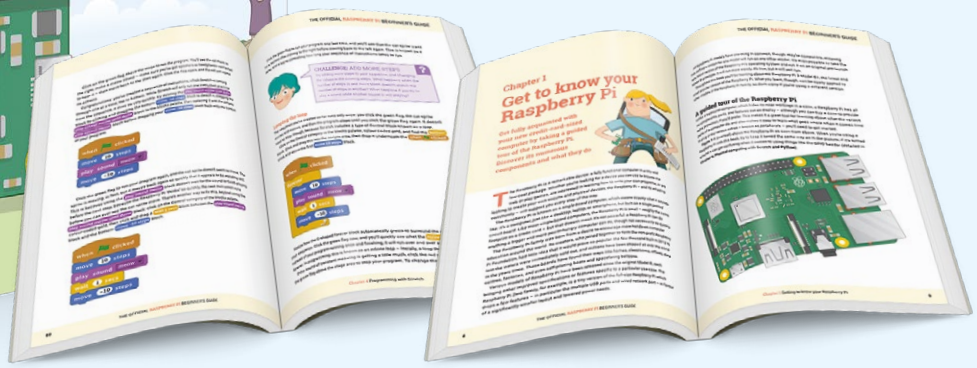
## The only guide you need to get started with Raspberry Pi

### Inside:

- Learn how to set up the Raspberry Pi, install an operating system, and start using it
- Follow step-by-step guides to code your own animations and games, using both the Scratch and Python languages
- Create amazing projects by connecting electronic components to the Pi's GPIO pins

**Plus much, much more!**

**£10 with FREE worldwide delivery**




**Buy online:** magpi.cc/BGbook

# Smart door

Adding a Raspberry Pi to your door has magical results. Want to see who's at the door or know when the post has arrived? Control the lock? Read on...

MAKER

**PJ Evans**

PJ is a writer, software engineer, and Raspberry Jam organiser. Too much of his house now thinks for itself.

**mrpjevans.com**

## You'll Need

- Raspberry Pi Touch Display **magpi.cc/touch**
- Camera Module **magpi.cc/camera**
- PIR sensor **magpi.cc/vmqYLG**
- 2 × Security door contact reed switch **magpi.cc/FDjbna**
- Wired doorbell **magpi.cc/KFYWcQ**
- PAM8302 amplifier **magpi.cc/mifFLc**
- Speaker **magpi.cc/PwkasX**
- Magnetic access control system **magpi.cc/rkEXYF**

Is your door a bore? Open and close, open and close. Snoozefest. Surely it can do more than that? How about a smart door that knows when someone approaches, when the post arrives, and can even offer remote viewing of the peephole? You can also add intelligent lighting, a controllable door lock, and facial recognition, all powered with your Raspberry Pi. So, let's ignore super-expensive door systems and build our own. You can do as much, or as little, as you like of this project and there's plenty of room for new and inventive uses.

## 01 Prepare your Raspberry Pi

Although you can use any WiFi-capable Pi, this is a perfect project for the new Raspberry Pi 3A+. Start by attaching the Pi to the Touch Display and preparing a microSD card with the latest Raspbian Stretch release. To allow easier access and mounting, we've detached the control board from the back of the screen, taking great care of the ribbon cable. Eventually, they'll be put in a smart 3D-printed case. Now, get your Pi set up and make sure to `sudo apt update && sudo apt upgrade` before proceeding.

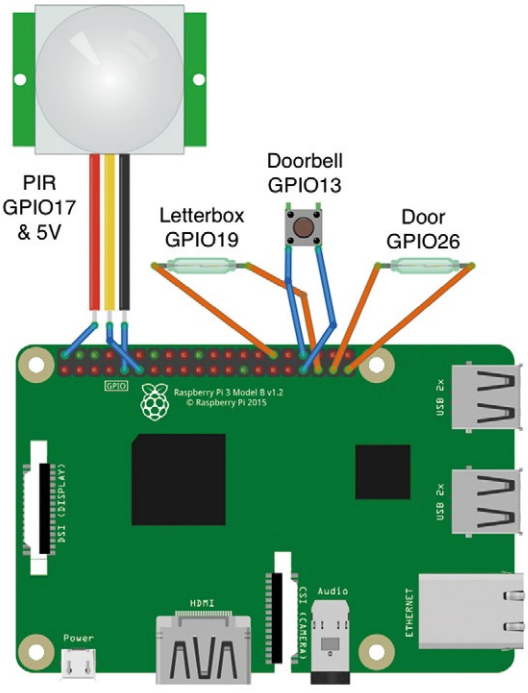
## 02 Attach the camera

We're going to keep an eye on the outside world by replacing the door's peephole with the Raspberry Pi camera. A peep-hole is typically a two-piece barrel that screws together and can be easily unscrewed from the inside. Remove the barrel and cover the hole with the camera. We're just going to affix this with tape for now; a printed mount will come later. Mount the screen and Pi to the door (we used 3M Command strips), placed so you can attach the camera's ribbon cable to the Pi once it's shut down. Make sure the camera is enabled in Raspberry Pi Configuration or raspi-config.

## 03 Footsteps approaching!

The first smart thing our door is going to do is detect someone approaching it. A cheap PIR sensor is perfect for the job. These cool little geodesic domes are triggered by heat and are the same gizmos that you find in motion-sensor lights, switches, and security systems. Connect to the Pi as shown in **Figure 1**, checking whether you have a 5 V or 3.3 V sensor. Sensitivity and duration of a 'detection' can be controlled by the two potentiometers on the PIR board. Mount this outside in a suitable location to 'watch' your door.

**Figure 1** The GPIO wiring that's needed for the various inputs and outputs



## 04 Monitor the door and letterbox

We have two magnetic reed switches, the type you find on windows and doors for security systems. They are made up of two parts: the wired part is a reed switch and the unwired a magnet. When the magnet meets the switch, it closes. If we attach the magnet to the door and the switch to the frame, when the door opens, so does the switch. There's no polarity to worry about, so connect one wire to GPIO 26 and the other to the adjacent ground. Repeat for the letterbox using GPIO 19. You may need a breadboard.

## 05 Ding dong!

Regular doorbells? Yawn. If we replace the doorbell with our own button, we can take a photo with the Pi Camera Module when someone presses it and send a notification. Way better. Mount a standard wired doorbell, which after all is just a momentary contact button, to the outside door frame and wire it back to the Pi using GPIO 13 and an available GND pin. If you're prototyping on a breadboard, a tactile switch will do fine.

## 06 Sounds good

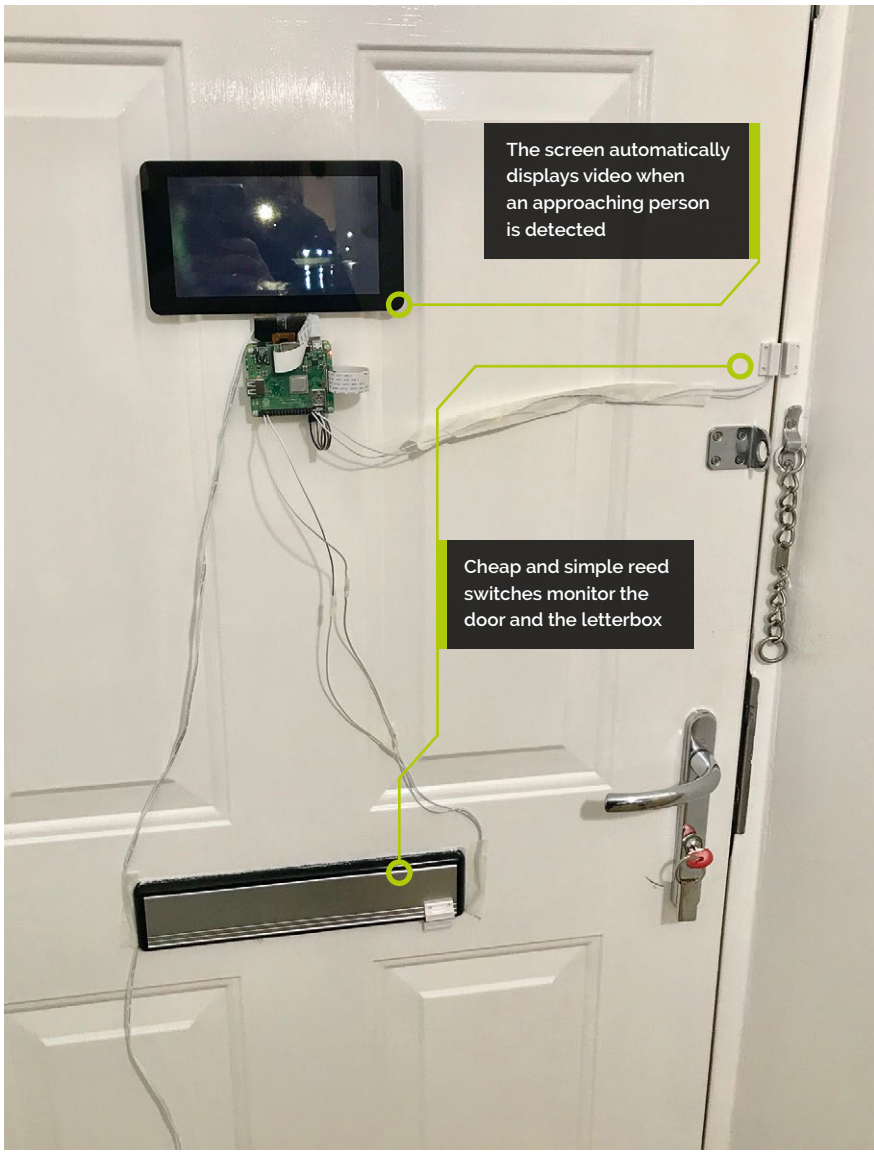
There's little point in a doorbell that makes no sound. We can use the small, but surprisingly powerful, PAM8302 amplifier with a speaker to make some noise. Supply power by soldering 'Vin' to an available 3V3 pin on the Pi, and ground to GND. To get an audio signal, you can tap the audio connector's signal and ground, then connect them to A+ and A- respectively. Finally, solder the speaker to the larger + and - terminals. When prototyping, you can skip this and use any active or passive speaker via the audio connector on the Pi.

## 07 Code

Double-check all your connections and power on the Pi. To use the code published here (overleaf), open a Terminal and enter:

```
mkdir ~/smartdoor
nano ~/smartdoor/smartdoor_test.py
```

Now type in the code as shown. Alternatively, to download all the code:


The screen automatically displays video when an approaching person is detected
Cheap and simple reed switches monitor the door and the letterbox

```
cd
git clone https://github.com/mrpjevans/smartdoor
```

To enable it to play our doorbell sample:

```
sudo apt install mpg123
```

Now test with:

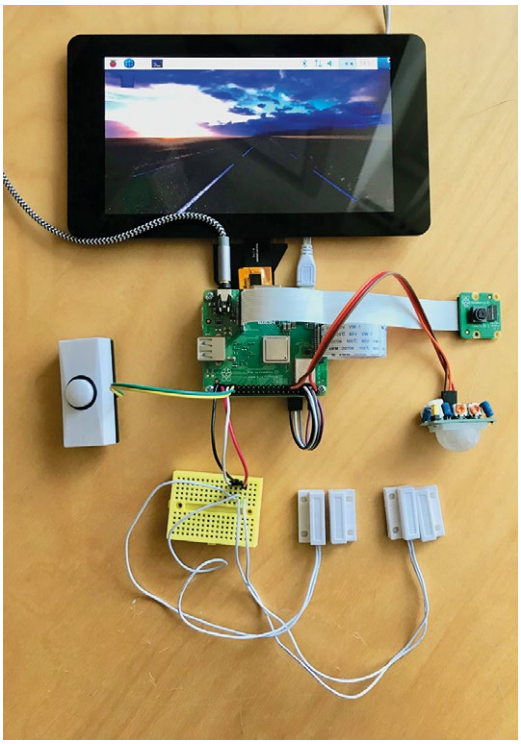
```
python3 ~/smartdoor/smartdoor_test.py
```

Watch the console output. If everything is working, you should be able to trigger the PIR, the reed switches, and the doorbell. The camera will capture ten seconds of video when motion is detected, and a photo when the doorbell is pressed. These are both saved to the desktop.

### Top Tip

**Night is dark**

If you want the camera to work well at night, you may want to consider a Pi NoIR Camera Module supported with some infrared lighting.



- You may want to prototype this project and test it before taking a drill to your door!
- The web app can run on the touchscreen, as well as on mobile devices or desktop browsers. Release the door from anywhere!

## 08 Get alerts!

Let's make this useful. Install Pushover on your phone, head over to **pushover.net**, sign up for a trial account, then log in and make a note of your User Key (a long string of characters). Now create a new Application and give it a name. Once created, you'll see an API Token. Make a note of this too. From the GitHub repository, edit **smartdoor.py** and add the User Key and API Token where shown. Run this version and you'll get phone alerts for each event and even a photo attachment when the doorbell is pressed.

## 09 Intelligent porch light

Following on from the Trådfri lighting tutorial in *The MagPi* #75 (**magpi.cc/75**), if you have an external porch light, why not make it smart! The file **porch.py** will connect a Trådfri smart light to an API that provides sunrise and sunset times for your location. Leave the script running and the light will switch on and off at the correct times. Additionally, it monitors the PIR sensor and will switch to full brightness when someone approaches! To use the script, get your latitude and longitude (you can use Google Maps or Earth) and edit **porch.py** as directed in the file.

## Top Tip

### Get the right lock

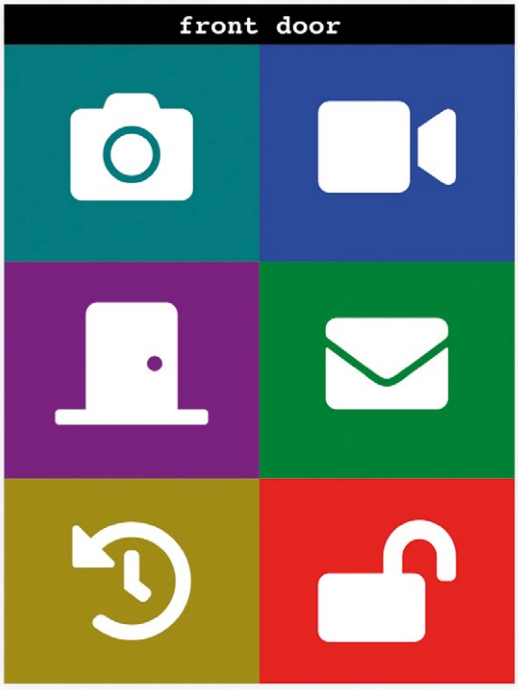
Magnetic door locks vary in size and shape; measure twice and order once!

## 10 Door lock

If you're interested in being able to control your door's lock, you may see that some solutions are very pricey. One that is perfect for experimentation is the magnetic hold lock, which uses an electromagnet to hold the door closed. The one we've used can withstand 180 kg of force, although stronger ones are available. The magnet mounts on the door and the electromagnet on the frame. The provided PSU contains a relay that can be powered by the Pi by simply connecting it to a spare GPIO line and ground. Please note this is no replacement for a proper door lock system.

## 11 Web app

If would be great to see what our door has been up to remotely, so a web app seems the next logical step. In the directory called **webapp** is a Python script that uses Flask to provide a web server that is usable on mobile devices. You can take a photo from the peephole, see the last recorded video, and even control the magnetic door lock from Step 10. Simply run the app alongside the others. Better still, set **smartlights.py**, **porch.py**, and **webapp/smartdoor.py** to start on boot (see the repository README).



## smartdoor_test.py

DOWNLOAD THE FULL CODE:
magpi.cc/dQyeQS

Language: **Python 3**

```
001. from picamera import PiCamera
002. from gpiozero import MotionSensor
003. from gpiozero import Button
004. from time import sleep
005. import os
006. import subprocess
007. import sys
008.
009. print('Getting smart...')
010.
011. # Set up all our devices
012. camera = PiCamera()
013. motion = MotionSensor(17)
014. doorSensor = Button(26)
015. letterbox = Button(19)
016. doorbell = Button(13)
017.
018. def motionDetected():
019.     print('Motion detected, video recording')
020.     os.system('DISPLAY=:0 xset s reset')  # Wakes
     the display up
021.     camera.start_preview()
022.     camera.start_recording(
     '/home/pi/Desktop/motion.h264')
023.     sleep(10)
024.
025. def motionStopped():
026.     print('Stopping video recording')
027.     camera.stop_recording()
028.     camera.stop_preview()
029.
030. def doorOpen():
031.     print('Door open')
032.
033. def doorClosed():
034.     print('Door closed')
035.
036. def letterboxOpen():
037.     print('You got mail!')
038.
039. def doorbellPressed():
040.     subprocess.Popen(['mpg123', '/home/pi/
     smartdoor/doorbell.mp3'],
041.                      stdout=subprocess.PIPE,
     stderr=subprocess.STDOUT)
042.     camera.capture('/home/pi/Desktop/doorbell.jpg')
043.     print('Someone\'s at the door!')
044.
045. # Attach our functions to GPIOZero events
046. motion.when_motion = motionDetected
047. motion.when_no_motion = motionStopped
048. doorSensor.when_pressed = doorClosed
049. doorSensor.when_released = doorOpen
050. letterbox.when_released = letterboxOpen
051. doorbell.when_released = doorbellPressed
052.
053. print('Smart door is smart')
054.
055. # Loop forever allowing events to do their thing
056. try:
057.     while True:
058.         pass
059. except KeyboardInterrupt:
060.     print('Smart door no longer smart')
061. except:
062.     print('Oh dear')
```

### 12 Facial recognition

Once a futuristic technology, decent facial recognition is now well within the grasp of the Raspberry Pi. Using the doorbell photo taken by the Pi, we can recognise a face using reference photos and send an alert to Pushover with the name of the caller! In a secure environment, a recognised face could even trigger the lock or you could play a welcome announcement. The install process is a little complicated, so if this interests you, see the documentation in the GitHub repository in the **face_recognition** directory of the 'smartdoor' repository.

### 13 Over to you

Here we've given you the basics to get going, but more complex events are possible. You could alert different people based on facial recognition or play custom doorbell tones. And, if you had problems with deliveries, video evidence can build up automatically. On a serious note, remember a lot of this is 'just for fun' and designed to inspire, so unless you're prepared to put in the work hardening the code and including failsafes, don't rely on this, or possibly make it as a fun kids' door project (but maybe without the lock!).

# Save it now!

Protect your precious data with backups, and stop snoopers with disk wipes

AUTHOR

**Richard Smedley**

Having often found words better than pointing at things, Richard stuck with the command line whence all around had fled.

@RichardSmedley

Your data is the most important thing on your Raspberry Pi, and good backups are a useful discipline to take elsewhere. Fortunately, the command line can actually make this easier.

The simplest method of all is to copy the data and move it elsewhere. It's labour-intensive, compared to automatic solutions, but for only-very-occasional backups it's better than nothing.

Whether you're copying to disk, or moving the backup to another machine, it's best to make it as small as possible. So you'll want to compress the file. We advise using the gzip compression format. You could go with xz or bzip2 compression (from tar, `J` or `j` options, respectively) for better compression to a smaller file size, but although alternatives to gzip save a little more space, they can take far longer to perform the compression.

Taking a directory of files that needs collecting together, then compressing it, can be done with a single tar command:

```
tar czvf mybackup.tgz myfolder
```

The `c` switch tells tar to create the archive, diving into all subfolders found; `f`, use the named file (here, **mybackup.tgz**), is necessary to direct the output away from the terminal. `v` asks for more verbose output, so the program tells you what it is doing and what (if anything) has gone wrong. Lastly, `z` invokes gzip compression. To unpack the archive, substitute `x` for extract in place of `c` – you can omit the `z`, as tar will recognise the compression type and automagically deal with it:

```
tar xvf mybackup.tgz
```

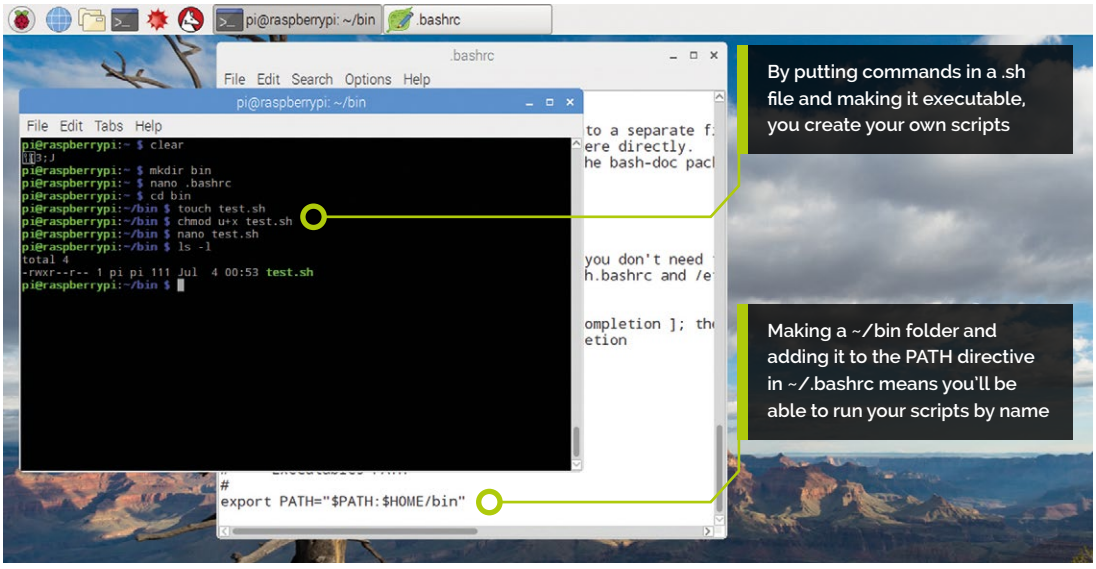
## A safe home

Often, using tar to back up **/home/pi** – with `cd /home`, then `tar` on the **pi** folder – will be all you need, but if you have data across directories, from **/etc** to **/var/www**, it's simplest to back up the entire microSD card. This can be done on the Pi, with the microSD card you want to back up in a

## Top Tip

### Tape archive

Tar dates from the days when computers backed up to big tape reels, those essential props of 1960s and 1970s sci-fi films. The lack of file structure on tapes means that tar can save all of the file system info such as ownership and timestamps.


pi@raspberrypi: ~/bin
.bashrc
File Edit Search Options Help
File Edit Tabs Help
pi@raspberrypi:~ $ clear
pi@raspberrypi:~ $ mkdir bin
pi@raspberrypi:~ $ nano .bashrc
pi@raspberrypi:~ $ cd bin
pi@raspberrypi:~/bin $ touch test.sh
pi@raspberrypi:~/bin $ chmod u+x test.sh
pi@raspberrypi:~/bin $ nano test.sh
pi@raspberrypi:~/bin $ ls -l
total 4
-rwxr--r-- 1 pi pi 111 Jul 4 00:53 test.sh
pi@raspberrypi:~/bin $
export PATH="$PATH:$HOME/bin"
By putting commands in a .sh file and making it executable, you create your own scripts
Making a ~/bin folder and adding it to the PATH directive in ~/.bashrc means you'll be able to run your scripts by name

USB card reader, with one important caveat: you'll be creating a file as big as the entire card – usually 8GB or more – onto a Pi with less space to spare. The solution is to compress the image file as it is created, which, for a Pi with a modest amount of data on it, will result in an image of around 2.5GB.

Use `sudo fdisk -l` before and after plugging in the card, to check how it is labelled. If **/dev/sdb** appears, say, with the size equal to the microSD card, unmount it with `umount /dev/sdb1`. Then back it up with the command:

```
dd bs=4M if=/dev/sdb | gzip > back-2018-12-04-raspbian.img.gz
```

Open another Terminal tab and monitor your disappearing disk space with `df` – if you don't think that it will fit, stop the dd operation with the usual **CTRL+C**, then `rm` (remove) the image file that you have partially created, and go and perform the backup on a computer with more disk space – or with a backup drive mounted, which you copy the archive to directly. Turning the backup into a usable microSD for the Pi means piping the other way, from gzip to dd:

```
gzip -cd back-2018-12-04-raspbian.img.gz | dd bs=4M of=/dev/sdb
```

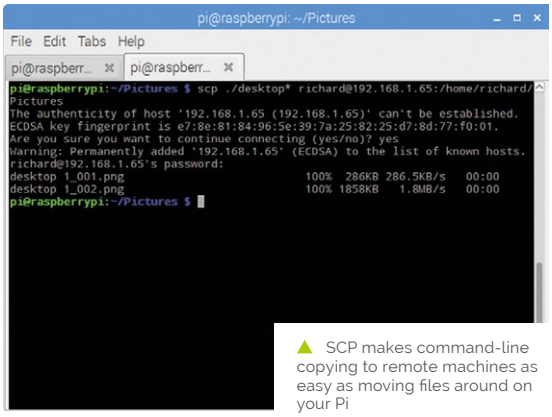
For disk operations like dd, you'll need root permissions: you can prefix `dd` with `sudo`, but for saving the file outside of **/home/pi** you may also need `sudo` – which means typing it in front of `gzip` as well.

This is only mildly inconvenient on the Pi, where `sudo` does not demand your password – but on a multi-user computer, or any setup with greater security, you need a reliable way of becoming the root user for every operation: running `sudo -s` will give you a shell with root permissions, but remember to exit afterwards. Alternatively, a chain of commands can be run with full admin permissions like so:

```
sudo bash -c "gzip -cd back-2018-12-04-raspbian.img.gz | dd bs=4M of=/dev/sdb"
```

## Remote copy

It's good to be able to make backups as required, using removable drives, but to move towards systematic backups you will need to copy across the network using SCP (secure copy protocol). To copy your backup file to another machine, one that allows SSH login (so is running a SSH server), pass your login name with the command:


pi@raspberrypi: ~/Pictures
File Edit Tabs Help
pi@raspberr...
pi@raspberr...
pi@raspberrypi:~/Pictures $ scp ./desktop* richard@192.168.1.65:/home/richard/Pictures
The authenticity of host '192.168.1.65 (192.168.1.65)' can't be established.
ECDSA key fingerprint is e7:8e:81:84:96:5e:39:7a:25:82:25:d7:8d:77:f0:01.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.65' (ECDSA) to the list of known hosts.
richard@192.168.1.65's password:
desktop_1_001.png 100% 286KB 286.5KB/s 00:00
desktop_1_002.png 100% 1858KB 1.8MB/s 00:00
pi@raspberrypi:~/Pictures $

▲ SCP makes command-line copying to remote machines as easy as moving files around on your Pi

```
scp -p back-2018-12-04-raspbian.img.gz pi@192.168.0.207:/home/pi/bak/
```

You will then be prompted for the user password. Change the `pi@` to whatever your user name is on the remote machine, not the Pi you're copying from. The `-p` preserves information such as when the files were last accessed. Note that `-P` (capital p) can be used to specify a particular port number.

Another Pi, with a static IP address and a plugged in USB disk drive, could be an inexpensive backup machine, as well as media server or whatever else your home or office needs.

Because you're sending these commands through the Bash shell, you get all the usual Bash advantages, from **TAB** completion (just type **bac**, or however much of the file name is unique in your present working directory) to wildcards. If you have disparate archives in the same directory – such as **www-backup-20181225.gz** and **data-backup-20181226.gz** – copy them all with:

```
scp -p ./*backup*gz pi@192.168.0.207:/home/pi/bak/
```

So far so good, but there are possibilities to automate your backup process later in this guide, so the interactive element – having to give a password – would be better avoided. As long as you can maintain security in some other way, of course.

## Key to logins

When setting up a SSH server, you can generate keys with `ssh-keygen` – these keys can be used to provide passwordless login. You can copy them across to other machines manually, but a handy shortcut is to use the command `ssh-copy-id`:

### Top Tip

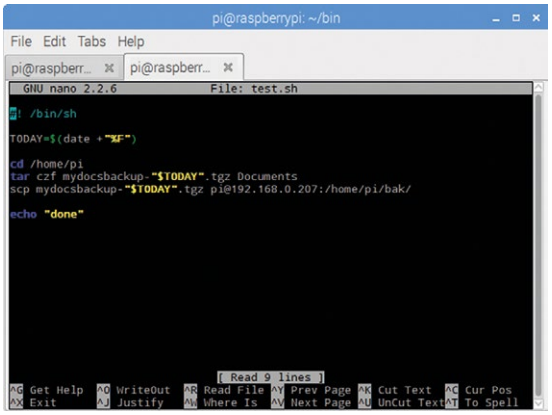
**Which files?**

Apart from ~, your project may have config files in **/etc** or even **/opt**, and under **/var** are logs, web config, and files – all of which you may have modified for a project.

### Top Tip

**Why remote?**

As well as being able to centralise backups for more than one machine, a remote backup protects you from unexpected disasters such as fire or flood where the Pi is. Fairly unlikely? Yes, but that doesn't stop you insuring your house.


pi@raspberrypi: ~/bin
File Edit Tabs Help
pi@raspberr...
pi@raspberr...
GNU nano 2.2.6 File: test.sh
#! /bin/sh
TODAY=$(date +"%F")
cd /home/pi
tar czf mydocsbackup-"$TODAY".tgz Documents
scp mydocsbackup-"$TODAY".tgz pi@192.168.0.207:/home/pi/bak/
echo "done"
[ Read 9 lines ]
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell

A datestamp in our script means that we are not producing a backup with the same name each day we run it

## Top Tip

### Hash bang

The shebang, or hash bang – #! – at the start of the script is an instruction to the program loader to run the program immediately afterwards (in the example in this guide, that's an interpreter directive to run **/bin/sh**) and pass the script as an argument to it.

```
ssh-copy-id pi@192.168.0.207
```

If you have more than one key pair, use `-i` to specify which .pub file you're copying. `-p` allows you to specify an alternate port number. Now we're all set for remote backups – but if you do them regularly, you'll waste a lot of disk space duplicating unchanging data.

Rsync lets you copy data in much the same way as SCP, but uses a delta-transfer algorithm to only transfer the difference between the copies of the source file on your disk and the remote, saved version. This both saves bandwidth used and avoids cluttering up your backup disk with multiple near-identical versions of a file. It's also handy if you're paying a cloud provider for storage and data transfer.

If your version of Raspbian doesn't have rsync, it's just an `apt-get` away. Typically, rsync uses SSH for transport, but you can set up a server running an rsync daemon, and directly contact the **rsync://** URL over TCP (defaults to port 873). In this case, set an `RSYNC_PASSWORD` environment variable or use the `--password-file` switch.

While rsync is not a built-in Bash command, we're highlighting it here as part of the array of command-line utility choices users face when considering whether to employ built-in commands or try something more complex instead. In addition, there is the possibility of using version control systems, such as git, for both backing up, and tracking changes on, important files.

## Top Tip

### Password free

Using your key to log in is a convenience you quickly get used to – beyond SCP to your backup server, try it on any machine under your control that you have to log in to.

## Script-it-yourself!

We have seen from early on how powerful Bash can be by chaining together a few commands; another way of putting commands together is to bundle them into a script – a short program simply comprising a small number of Bash commands, and known as a shell script. Take a look at this code – try typing it in to your favourite text editor, adjusting it for the IP address of your networked backup server, and backup folder location (or change the `scp` line to a `cp` to a plugged-in backup drive), and saving it as **test.sh**.

```
#! /bin/sh
cd /home/pi
tar czf mydocsbackup.tgz Documents
scp mydocsbackup.tgz pi@192.168.0.207:/
home/pi/bak/
```

Then make the script executable with:

```
chmod u+x test.sh
```

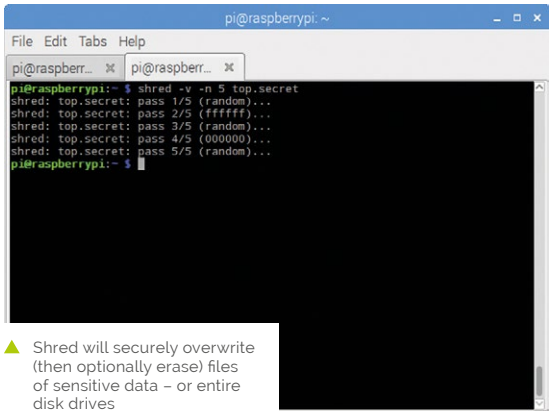
...and run it with `./test.sh` – any problems, then check the names, network address, and did you perform the `ssh-copy-id` step? Now we have a script that saves a folder, and copies it remotely, do you notice any potential problems? Each time you run it, it will overwrite the previous **mydocsbackup.tgz**, both locally and remotely. We need a way to put a timestamp on the backup name:

```
#! /bin/sh

TODAY=$(date +"%F")

cd /home/pi
tar czf mydocsbackup-"$TODAY".tgz
Documents
scp mydocsbackup-"$TODAY".tgz
pi@192.168.0.207:/home/pi/bak/
```

What we have done is set a variable – `TODAY` – to the current date, in YYYY-MM-DD format, which we can now access with `$TODAY`. You can run `date +%F` in the terminal – `date --help` will show you the many other format options. Now you can automate it by putting the script somewhere like **/usr/bin** (and with a better name than **test.sh**), and running it regularly with cron, as we covered in *The MagPi* #76 'Start and stop at your command' guide.

Shell scripts tend to grow; there is always room for improvement. Here, you may want to back up more than one directory, for example, or use `echo` to let users know what the script is doing at each stage. You could even make it interactive, letting users choose which directories to back up.

There are plenty of shell scripting tutorials online, and great books to take it further, but Raspbian itself holds many great shell scripts, from which you can learn. To see how a script can be organised to still be maintainable with over 1,000 lines of code, have a look at **/usr/bin/raspi-config**.


pi@raspberrypi: ~
File Edit Tabs Help
pi@raspberr... pi@raspberr...
pi@raspberrypi:~ $ shred -v -n 5 top.secret
shred: top.secret: pass 1/5 (random)...
shred: top.secret: pass 2/5 (ffffff)...
shred: top.secret: pass 3/5 (random)...
shred: top.secret: pass 4/5 (000000)...
shred: top.secret: pass 5/5 (random)...
pi@raspberrypi:~ $

▲ Shred will securely overwrite (then optionally erase) files of sensitive data – or entire disk drives

However grand or modest your scripting ambitions, don't be afraid to try things out: build up gradually, and test your code each time, so you know where to look for any errors you introduce. Help is at hand from the Raspberry Pi forums, and pasting your code into **shellcheck.net** will give you valuable feedback – for example, advising you the `cd` line of our backup script should be:

```
cd /home/pi || exit
```

…in case the `cd` step fails – this is generally a good idea, although not so important in this particular case. Now that we have a choice of backup options, one task remains: securely getting rid of data from disks. This is a concern for anyone handling other people's data, or just protecting their own privacy and security.

## Through the shredder

Back in the 1960s, if you wished to cover your tracks, *Mission Impossible* told us it was done by the show opening's taped mission assignment finishing, "This message will self destruct in ten seconds…", and boom went the tape player.

In these digital days, protecting privacy or security means understanding how a disk drive actually stores data, so that you don't dispose of old disks under the mistaken impression that you've securely erased data, when you haven't.

Disk space is collected into blocks, sized typically at 4096kB, that are indexed by the file system with inodes so that the disk controller knows where to send the read head to retrieve information. SSDs and flash drives don't have read heads, but still organise the data in a similar fashion. Most disk operations occur at the inode level – so moving a file between directories on the same disk partition is simply done by relabelling an inode. `rm` does not delete the data stored, just the reference to its blocks on the inode.

Plug in a disk drive after someone has done `rm -rf` on it and it will look empty, but use a low-level utility and you'll have access to all of the jigsaw puzzle pieces needed to put the data back together again. So far, so *NCIS*, but can this be significant to the average Pi user? Given the range of projects out there, and the multifaceted data that they collect, to stay on the right side of new and future data protection laws it will be useful to know how to securely remove data from your disks.

## Caution

Before we start erasing disks, think of the carpentry maxim, 'measure twice, cut once'. It's easy to erase the wrong disk or partition if you're not paying attention. Given enough opportunities, most of us do it, and it's often the lesson that teaches us to make proper backups! Running through other operations on the disk (`mount`, `df`, `ls`, `umount`), before erasing, works as a sanity check that you're addressing the correct partition.

Now to those blocks. You can overwrite every bit of information, either with all 1 digits, or with random data, using `dd`. Even then, with magnetic disks, it's theoretically possible to recover the data, and multiple overwrites will be necessary – but before you worry about writing a script to do that, let us introduce shred, a utility that does just that, overwriting with as many passes as you select as a command switch (or defaulting to three):

```
shred -vf -n 5 /dev/sdb
```

Adding a `-z` switch will overwrite the random data shred has used, with zeroes, leaving a new-looking disk. `-u` will delete the file after the secure overwrite. Shred can also safely overwrite and/or remove individual files.

## Top Tip

### Where am I?

If you follow our tip on SSH keys everywhere, and end up hopping from machine to machine, remember to customise your Bash prompt so that you are always sure on which machine you're about to erase a file.


## Conquer the Command Line

For more terminal tutorials, take a look at our Essentials book, *Conquer the Command Line*. Discover the power of the command line to simplify complex tasks, or instantly carry out simple ones.
**magpi.cc/Essentials-Bash**

Part 12

# Coding games on the Raspberry Pi in C/C++

Let's shoot something by adapting our code to another genre

MAKER

**Brian Beuken**

Very old game programmer, now teaching very young game programmers a lot of bad habits at Breda University of Applied Science, in the Netherlands.

magpi.cc/YxaUVQ

## You'll Need

- Code::Blocks `sudo apt-get codeblocks`
- GLM
- stb_image.h
- TilEd or similar map editor

## Top Tip

### Be creative

Rather than dotting enemies around, give them spawn sites and use a pattern in their actions!

Time to wrap up a year of solid learning with a review of all the things you didn't know you were learning, and see how we can take them a little further.

## Consider what we did!

It might not seem like it, because we could only do a few things each month, but we have in fact covered rather a lot of concepts, and implementing them in code has exposed us to some of the core fundamental skills any aspiring C++ game programmer needs to write games.

There's still quite a long way to go to be at the triple-A lead coder level, but that will come with time and practice – exposing yourself to more advanced concepts and more C++ methods, as and when you need them.

For now, though, let's really think about what we have given ourselves.

**Variables:** We know we can manipulate variables, and that there are many types of variables.

**Flow control:** We saw that our project flows from one instruction to the next, but that condition tests allow us to change the direction of our code between two options.

**Loops:** We can create a loop to repeat a process until it's done, or until a certain number of steps have been taken.

**Object control:** We took our first real steps into the concepts of OOP (object-oriented programming) by collecting different bits of data into one overall object, which we were able to manipulate by changing some of the object's variables.

**Abstraction:** We saw that by thinking of things in terms of an object, rather than as a collection of variable sets, we can visualise that data set as a thing that does something... as we ask it to.

**Visualisation:** Although we've only scratched the surface, we've been able to understand that our CPU and GPU do different things, and setting up our GPU to visualise things in different ways is a major achievement.

**Maths:** Yes, we did some maths, and hopefully we saw that sometimes using more complex maths can be much easier than doing a long series of simpler equations.

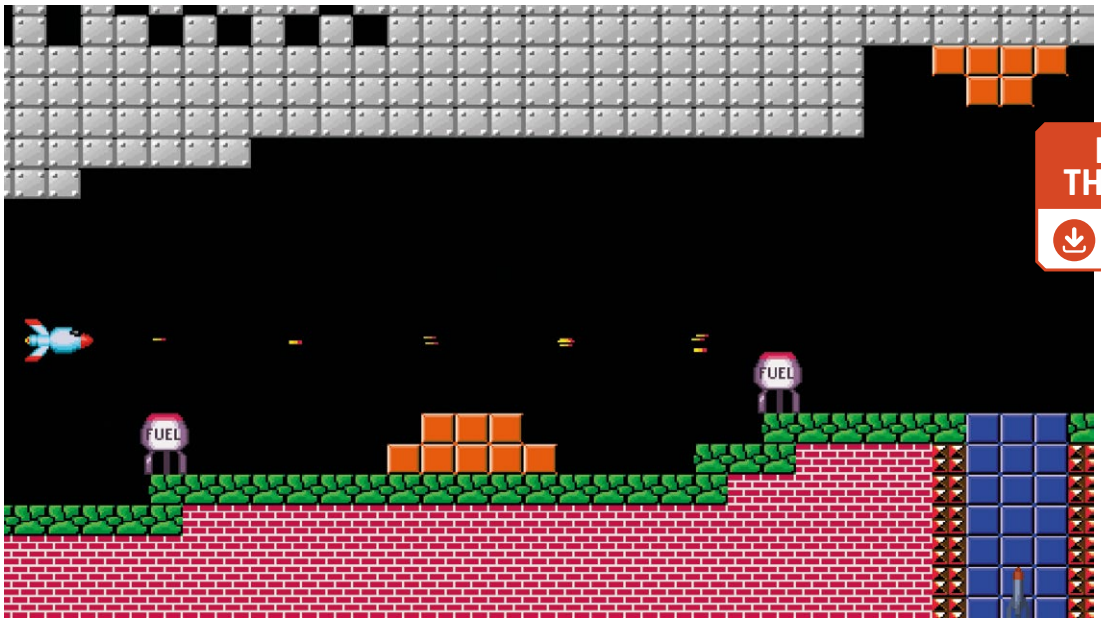
**Graphic handling:** We also managed to isolate the idea that an object's graphic isn't the object, but simply a visual representation of a thing. This makes it easier to conceptualise.

**Data representation:** We not only managed to encapsulate custom data sets into our objects, but we also made use of some standard C++ data systems, like arrays and vectors.

**Libraries:** A key point of C++ coding is being able to use code written by much smarter coders to perform complex things for us. We can print text on screen and use maths systems which might be way beyond our beginner's levels of comprehension.

**A core set of C++ instructions:** Using just maths, conditional tests, OOP principles, and very simple GPU, we were able to produce a nice 2D platform game which was simple to change to 3D(ish). The mantra of 'simple code works best' is something we should try to remember even when we do finally branch out to the more complex C++ features; it will make some of our tasks that much easier.

**Problem solving:** By far the most important thing is that we started to view coding as a sequence of problems to be solved in logical ways.




**DOWNLOAD THE FULL CODE:**
magpi.cc/eCjwdf

**Figure 2** Pass the ammo, keep spitting out bullets, but remember to remove them as well!

All of these points and a few more are now on show in your fully working (semi) 3D-viewed platform game.

And to prove that it is all the same code in this final lesson, we'll use the same principles to create a small scrolling shooting game using almost all the same code and some new code.

As before, we'll define a base graphic object, make a map, add some code to create objects in the map, update them, and let our player – this time a ship – travel through the map.

## Step by step

Even though we have code from previous projects, we'll start a new project, but copy the files from the platform game as we need them. C++ class files should be as transportable and complete as possible. There will be a few things to tweak, but that's fine.

We need a map, so all the map code is pretty usable with a few tweaks for flexibility. We will even use the same graphics, but arrange them differently.

We require the basic concepts of the SimpleObj again.

We need a player, in this case a ship, so all new graphics have been supplied for moving objects.

We need a camera, and we'll keep a slight sense of 3D visualisation, as before, by moving the camera around a little bit to show the angles; however, we're not doing a full 3D game.

We need enemies… well, we're not doing quite the same enemies, but many of the code concepts are the same.

So, we've got enough to work with. The biggest difference between this game and a platform game is that we don't need gravity: we assume the ships are all flying. We will make the scroll constant, so that the player ship stays central and also moves forward with the scroll. That is enough to get going.

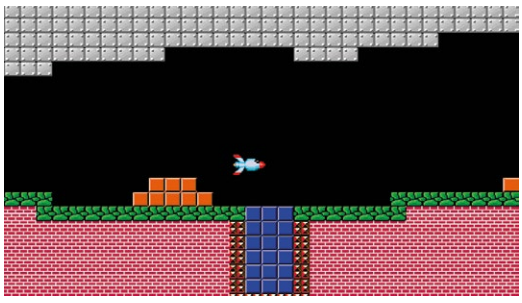


**Figure 1** Reuse the graphics but with a new long map

## Flexibility, let's have some?

One other important feature this time: we've made things much more flexible. Hard-coded, fixed-sized arrays are effective up to a point, but suppose we need different-size maps, with different graphics? Arrays limit us by requiring us to use the same size each time, but we know vectors are more variable, so let's use them.

Even though we're not using a lot of textures, we don't really want to be loading the same texture two, three, four, or however many times.

The same applies to shaders. Internally, a shader is nothing more than a little text file which

### Top Tip

**Loading is better**

Time to move away from hard-coding things: being able to load .txt files lets us load all kinds of things and change them without compiling.

### Top Tip

**KISS**

'Keep It Simple, Stupid!' is a coding mantra. Abstract away the hard work so you only need to think what you want to do. Simple code nearly always works best.

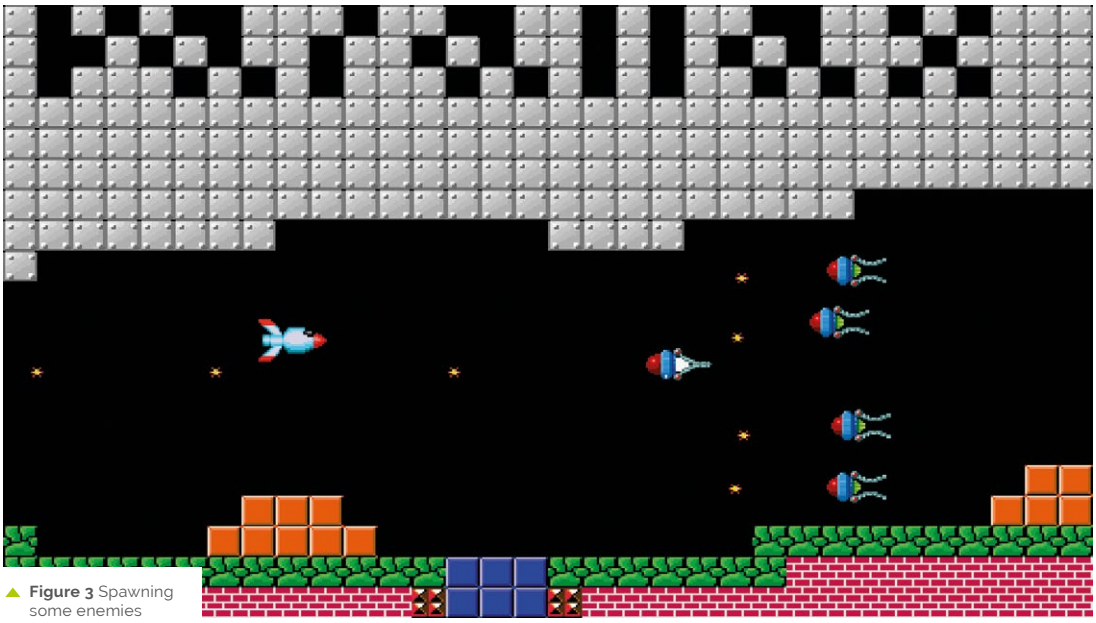
NEW

# Wireframe

Join us as we lift the lid on video games

NEW £3
ALL FORMATS
Wireframe
LIFTING THE LID ON VIDEO GAMES
Getting started with Unity
How to make a first-person shooter
Doom to Dota 2
When modders turn professional
Thrunt XL
An indie game made without a line of code
SCARE TACTICS
X-COM'S JULIAN GOLLOP ON HIS NEW GAME, PHOENIX POINT
EXCLUSIVE
Far Cry 4's Alex Hutchinson on his "louder, brasher" game
Mega Man
The changing face of a console icon
Solid Snake
Metal Gear creator Hideo Kojima profiled

Visit **wfmag.cc** to learn more



▲ **Figure 3** Spawning some enemies

is parsed and compiled. We know how to load things, so let's just load these little .txt files when needed.

These kinds of things are prime candidates for a bit of housekeeping. We can use simple STL systems like 'map', to allow us to load data from our stores and keep track of them only using what we need. Map is like an array, but instead of a numerical index to get our value, we can use a string to associate a file name with a value we need.

> Game programming is especially concerned with performance: we should consider ways to avoid having 300 not yet visible objects

You can see three new files/classes: ShaderManager, TextureManager, and LevelManager. They contain classes which will load things for us, when we need them, and keep track of them when we want to use them. All three new classes have explanations of what they do and why.

Now, let's get a new game written.

## Top Tip

### Make the game playable

As much as we can flood our game with hard-to-kill baddies and bullets, good games should never be so hard as to be impossible.

## RocketBob

Bob was our platform game's main character and we'll use a main 'character' again, but this time Bob is going to be a rocket ship – see **Figure 1**.

Creating a new RocketBob class is simple enough: all we need to do is think about what actions we want the rocket to do. Mainly, that's going to involve moving forward at a steady pace, and up and down. We can also allow left and right movement, but only in addition to the constant forward motion.

'He' should shoot: we can't have intergalactic warfare without a few bullets. We should also make sure he can't move into the solid areas of our map, which for this game will be a short tunnel filled with marauding waves of enemy ships, many of which shoot at us or attempt to ambush us.

Now we know what he can do, the class is easy to create, and can be seen in the source code.

## The subjects of our aggression

Enemies are going to mostly follow the same rules as platform game baddies, and collision with them is deadly, but we need to consider how we make them. Should we create them all at the start of a level, or should we create them as we need them?

This distinction is important: we have a fairly small test map, so we could get away with defining 100, 200, even 300 enemies at the level initialisation and updating them even when they are not visible on our screen, but it's an awful waste of processing time. We might only see 10–20 of them on screen at any given part of the map and what if you decide to create an even bigger map? Populating that map could take some time. We might need to place certain things at fixed points, though – see **Figure 2**.

Game programming is especially concerned with performance, and so we should consider

ways to avoid having 300 not yet visible objects going through their updates and draw calls for no effect.

But suppose we only had ten special objects or test points whose job is simply to trigger a wave of baddies once RocketBob is close enough to make it worth doing. That's a much easier proposition.

Creating Trigger objects means we can limit their update to testing if RocketBob is close, and if so, the Trigger itself can take responsibility for creating waves of baddies, before shutting itself down. See **Figure 3**.

We've made a set of trigger classes, each of which can set in motion a specific wave of baddies or missiles at the right time and place. All we needed was a simple range test and the ability to create sequences of baddies from a simple list. Again, review the code.

Our enemies themselves are not especially complex – the game difficulty comes mainly from the numbers of them active at a time and their relentless bullet shooting. There should be quite a few of them on screen, and predictable spawn patterns can help our player to blow them away before they get a chance to shoot. See **Figure 4**.

## Pass the ammo

Bullets are also a simple type of object, easy to create when we press a fire key. We can have two types – dumb and targeted – and we'll try to create some variety with that. For targeted bullets, we use a vector between the shooter and its target to find the direction the bullet needs to travel, and turn that in to a unit vector which gives us consistent direction for the bullet. Because it's a unit vector (i.e. if we made it a triangle, the hypotenuse would be one unit long), we can increase the velocity of the bullet by multiplying by a scalar value. So different ships can have different-speed bullets.

## More power

As with our platform game, we'll litter the level with a few power-ups, which we will use to create different firing states and perhaps create a few cool homing bullets. We could place them with triggers, but as they are rather simple singular objects, a standard placement is fine.

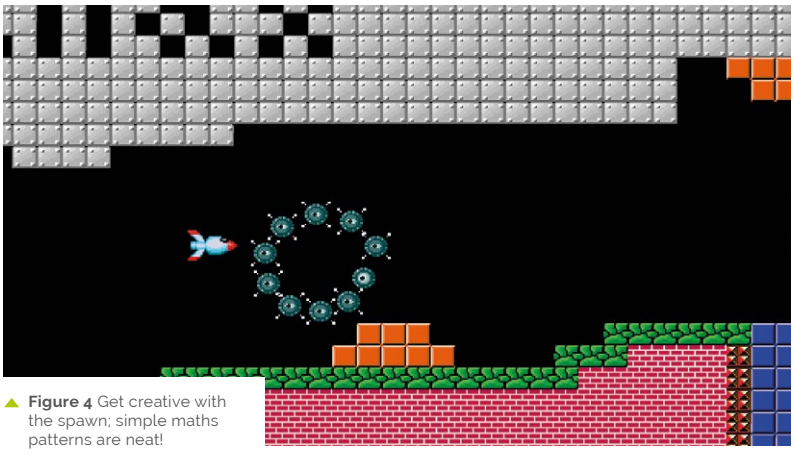


**Figure 4** Get creative with the spawn; simple maths patterns are neat!

## And we're done

That's it! Some simple logic for the baddies and a simple change of main character with a scroll that edges ever onward, and we have a totally different game. But this is the important part: using the same basic concepts of a simple object being used to derive various different objects which interact with an environment.

## Taking this forward

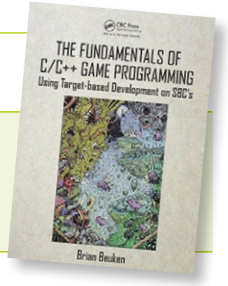
Like our platform game, we could go to 3D with this; that isn't a massive challenge, but interaction in 3D space presents more problems than visualisation, and requires a better understanding of maths and also an awareness of some performance issues with our Raspberry Pi.

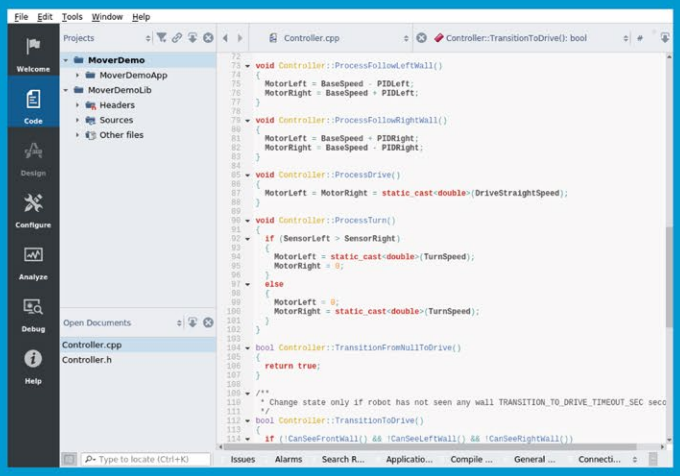
For now at least, focus on 2D concepts until your programming skills feel ready for the new challenge. There's a wealth of retro-style games you can produce using this simple framework, which you can build up and develop into your own game engine. And when you are ready, there are dozens of good sites to take your first real steps into 3D.

That's all for these lessons in core skills. We hope it has increased your confidence and given you a taste for C/C++ coding to make your own games.

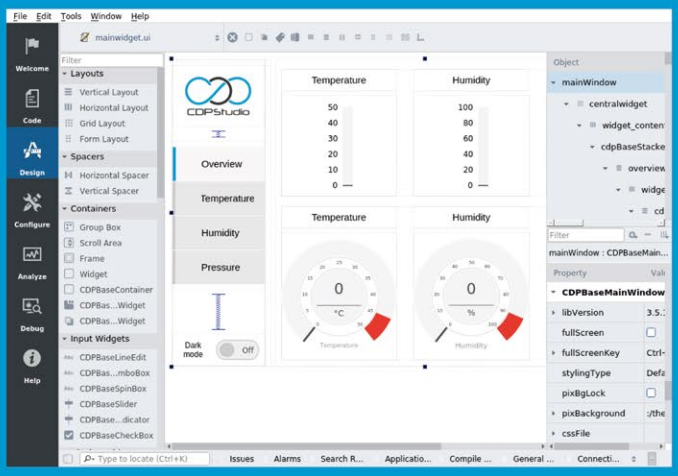
## Learn more about C

Brian has a whole book on the subject of making games C and C++, called *The Fundamentals of C/C++ Game Programming: Using Target-based Development on SBCs*. Grab it here: **magpi.cc/nUkjEt**

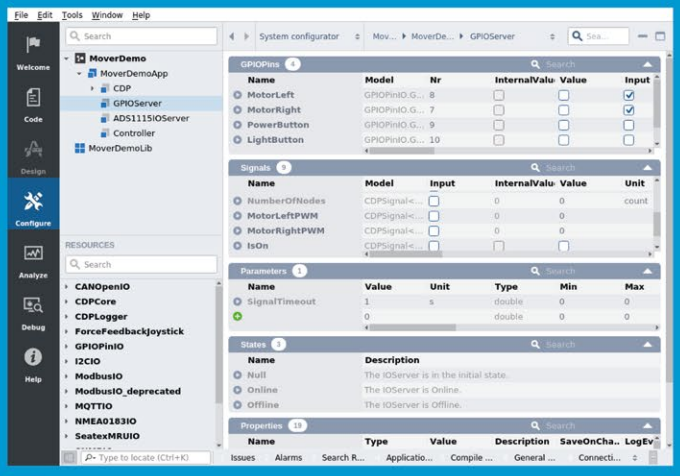




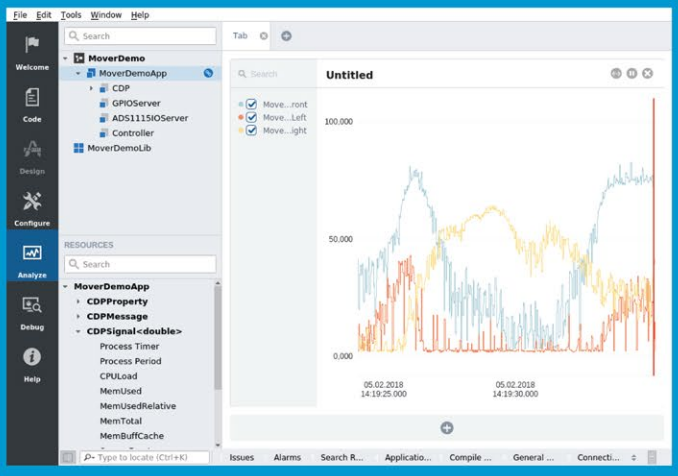
Code



Design



Configure



Analyze

# Now free for home projects

## A professional control system development tool

CDP Studio is a development platform for industrial control systems, now coming with a free version for non-commercial use. The system can run on a Raspberry Pi, supports C++, open source libraries and has a large feature toolbox including GPIO, I2C and MQTT. Its built in GUI design tool and features lets you code less and do more.

Free download on www.cdpstudio.com

CDP Technologies AS
Nedre Strandgate 29
P.O. Box 144
NO-6001 Ålesund, Norway

Tel: +47 990 80 900
info@cdptech.com
www.cdpstudio.com

CDPStudio

THE Official

# RASPBERRY PI PROJECTS BOOK

VOLUME 4

## 200 PAGES OF IDEAS & INSPIRATION

DIY Games Console

Build a Robot

Make a Magic Mirror

Set up a Spy Cam

**55 PROJECTS & GUIDES**

FROM THE MAKERS OF The MagPi THE OFFICIAL RASPBERRY PI MAGAZINE

THE OFFICIAL RASPBERRY PI PROJECTS BOOK VOLUME 4

200 pages of ideas & inspiration

£12.99
200 pages of
Raspberry Pi

THE Official
# RASPBERRY PI PROJECTS BOOK
VOLUME 4

Amazing hacking and making projects
from the makers of The MagPi magazine

Inside:

- How to get involved with the Pi community
- The most inspirational community projects
- Essential tutorials, guides, and ideas
- Expert reviews and buying advice

Available now magpi.cc/store

plus all good newsagents and:

WHSmith BARNES&NOBLE

Available on the App Store

GET IT ON Google Play

# THE BEST RASPBERRY PI ACCESSORIES

## Our guide to the greatest add-ons, HATs, and expansions for your Raspberry Pi

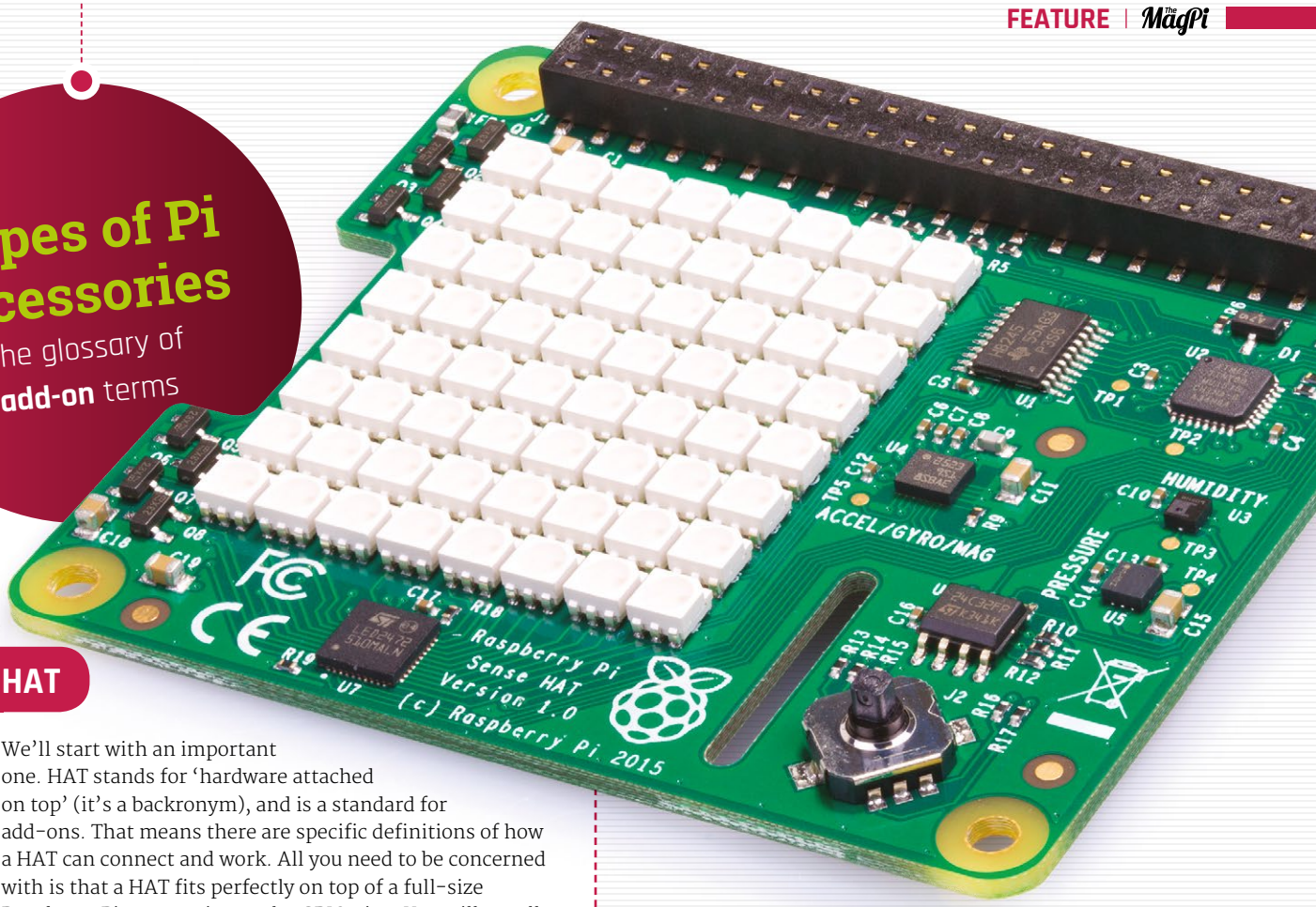
The Raspberry Pi is fantastic on its own. With wireless LAN, Bluetooth, USB ports, the multitalented GPIO pins, and a full operating system, it really is a fully functional computer.

What if you could make it better? Do more? Make certain functions easier? This is where Pi add-ons and accessories come in – there are hundreds of extras you can get for your Raspberry Pi that will improve its already impressive specs.

What add-ons should you get, though? We're here to show you the best the Raspberry Pi world has to offer.

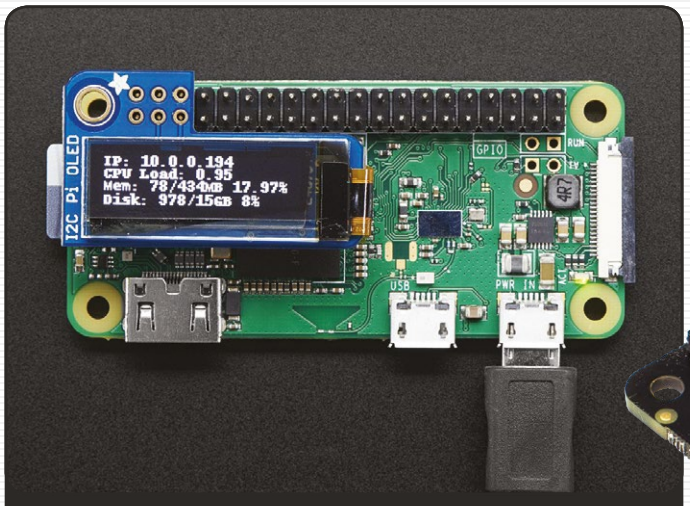
# Types of Pi accessories

The glossary of **add-on** terms




## HAT

We'll start with an important one. HAT stands for 'hardware attached on top' (it's a backronym), and is a standard for add-ons. That means there are specific definitions of how a HAT can connect and work. All you need to be concerned with is that a HAT fits perfectly on top of a full-size Raspberry Pi, connecting to the GPIO pins. You will usually need to install extra software to get them working.

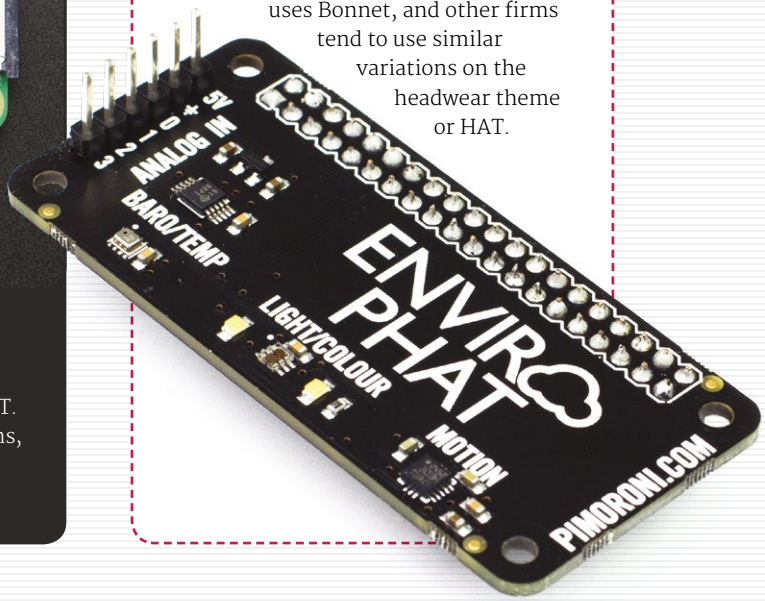



## Add-on

A HAT is an add-on, but an add-on isn't necessarily a HAT. Add-ons can be connected via USB, only specific GPIO pins, the camera port, etc. It's a catch-all term for something that you can add onto your Raspberry Pi in some way.

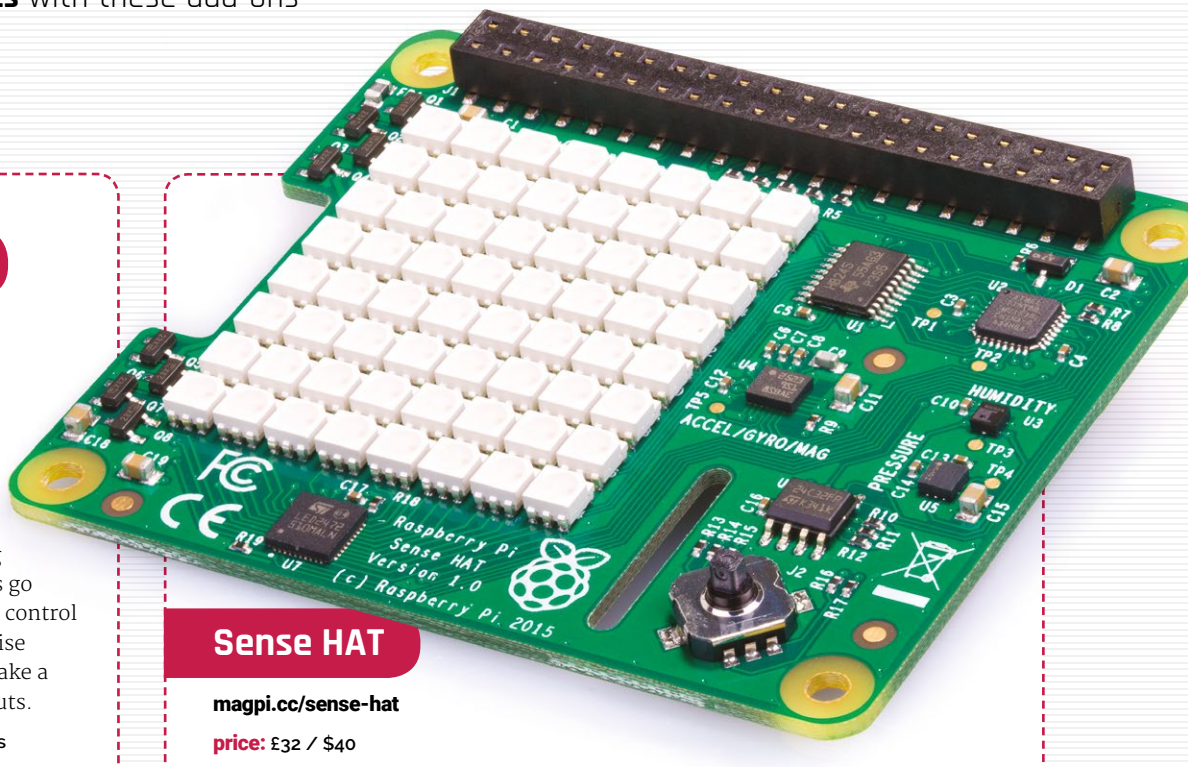
## pHAT/Bonnet/Cap/etc.

While the HAT standard has recently been extended to different sizes, many companies have created their own term for add-ons that act like a HAT but don't quite meet the technical criteria. Pimoroni coined pHAT, and Adafruit uses Bonnet, and other firms tend to use similar variations on the headwear theme or HAT.



# PROGRAMMING AND MAKER ACCESSORIES

Improve your **projects** with these add-ons




## Sense HAT

**magpi.cc/sense-hat**

**price:** £32 / $40

Developed for the Astro Pi mission, the Sense HAT is primarily a set of environmental sensors. It can detect pressure, humidity, temperature, and orientation/movement. It also has an 8×8 LED matrix and a little joystick for very basic display and control. There are a couple of Sense HATs running on Raspberry Pi boards up on the ISS at the moment.

**Use it with:** Weather stations, space stations

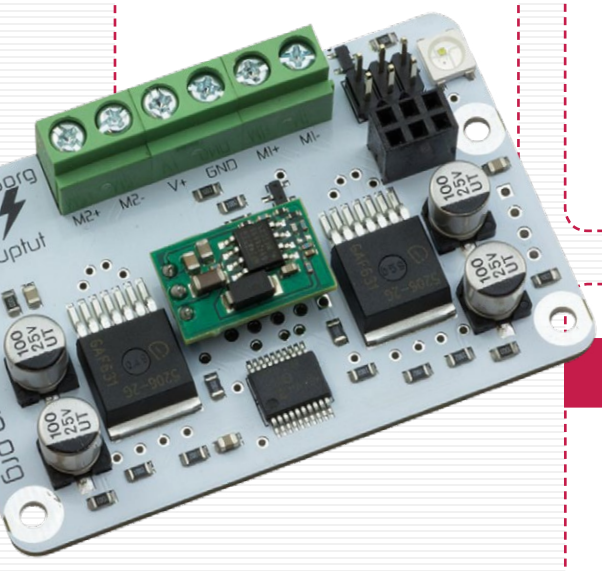
## ThunderBorg

**magpi.cc/xcsSpm**

**price:** £29 / $37

This dual motor control board by the folks at PiBorg is a powerful and robust way to use a Raspberry Pi to control a robot. As well as being able to make the motors go forward and back, it can control the speed for more precise manoeuvring, and can take a wide array of power inputs.

**Use it with:** Robots, RC cars




## Hologram Nova

**hologram.io/nova**

**price:** From £50 / $64

Despite the world's best efforts, WiFi is not available absolutely everywhere yet. You can't even plug an Ethernet cable into a tree! The Nova solves this problem by letting you connect to cellular networks around the world on your Pi via this simple USB device.

**Use it with:** Portable projects, outdoor cameras

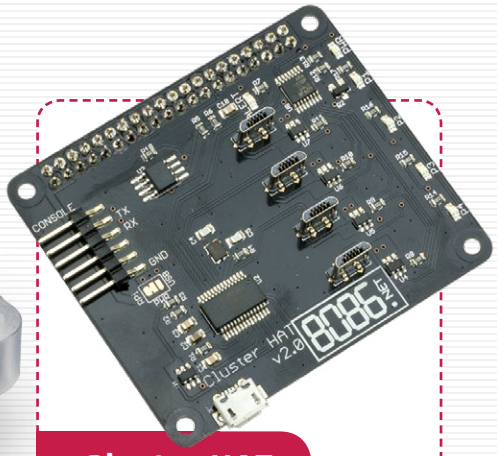
## Edge TPU Accelerator

**magpi.cc/TFLQpM**

**price:** TBC

Google was responsible for the AIY Voice Kit we gave away with issue #40 of *The MagPi* – a special HAT and extras that lets you turn your Pi into an AI voice assistant. The Edge TPU accelerator aims to help the Pi out with AI tasks, and make your voice assistants better.

**Use it with:** Voice control, AI projects




## Cluster HAT

**magpi.cc/Qrshbi**

**price:** £29 / $37

Putting Raspberry Pi boards into a cluster isn't too difficult, but the Cluster HAT makes it very easy indeed. Using one Raspberry Pi, you can distribute computing to up to four Pi Zeros. Cluster computing is a great way to perform different kinds of calculations.

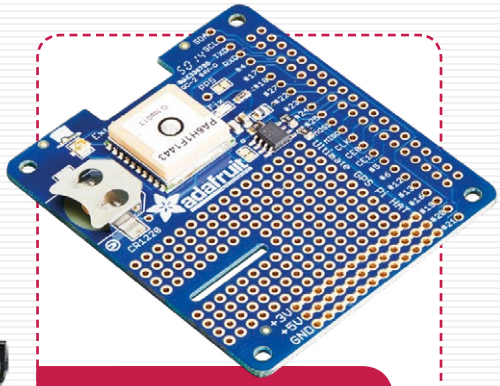
**Use it with:** Mathematics, big algorithms

## GFX HAT

**magpi.cc/ZWvcLG** **price:** £20 / $25

A very functional screen that draws a lot less power than most displays, it also includes six capacitive touch buttons for control. It's the successor to an excellent add-on known as Display-O-Tron, and makes interactive projects look great.

**Use it with:** Thermostat, practical projects




## Ultimate GPS HAT

**magpi.cc/vmPqLL**

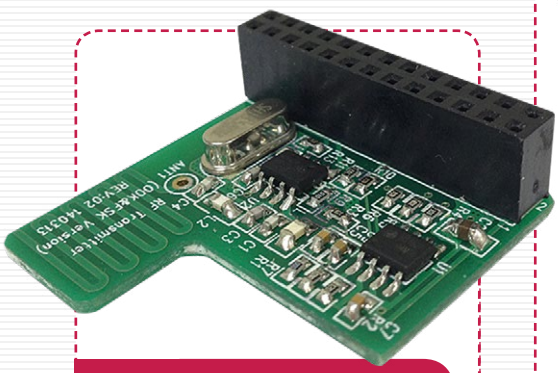
**price:** £29 / $37

This GPS add-on allows you to track the precise location of your Raspberry Pi – well, as precise as GPS can be. You'll have to combine the position data with map data to make much use out of it, but once you have it all working it's pretty fantastic.

**Use it with:** Sat-navs, route tracking

# INDUSTRIAL AND POWER

Using a Raspberry Pi for **enterprise?** You may want to check these out...

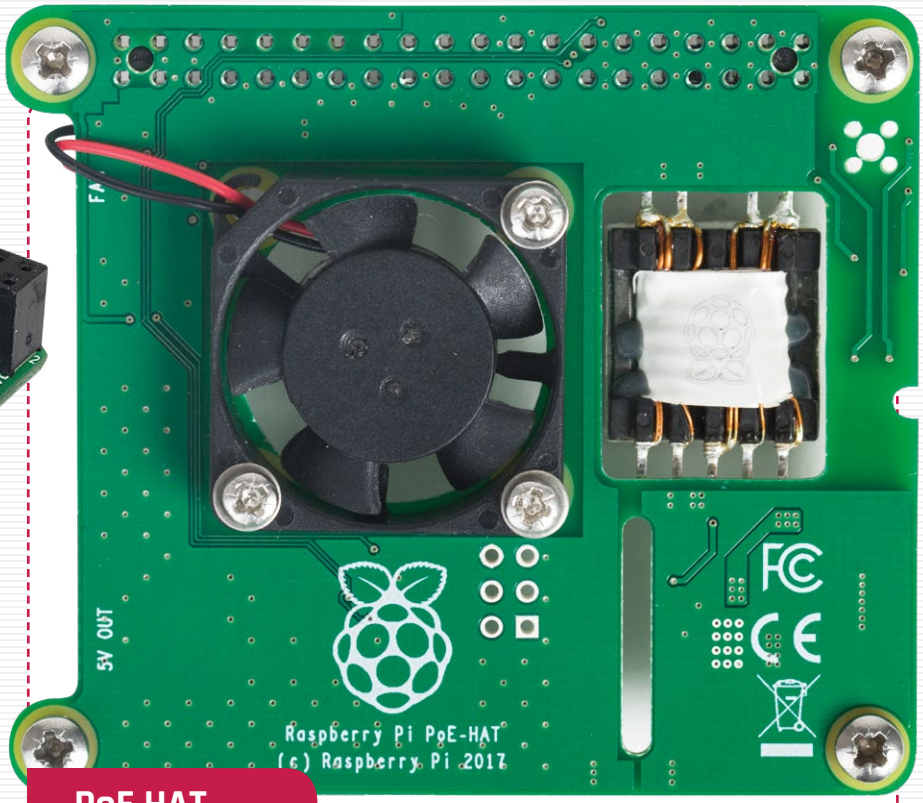


## Energenie Pi-mote

**magpi.cc/FnezCP**
**price:** £22 / $28

Another remote-control power solution, this one is designed for the Pi to control power elsewhere. There are various Energenie Pi-mote products, the cheapest starting at £10, but the starter kit version also includes compatible sockets. You can even get a Pi-mote that allows the plugs to talk to the Pi as well.

**Use it with:** Timed-lighting, some IoT

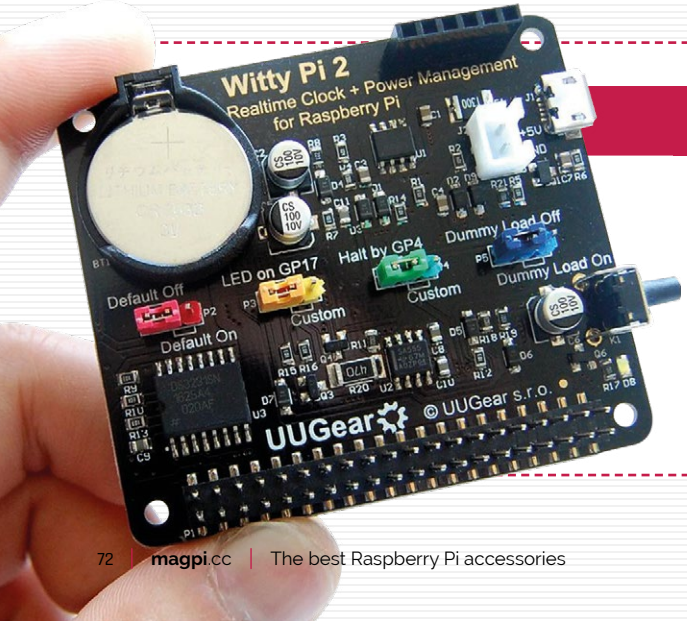



## PoE HAT

**magpi.cc/aqpwZc** **price:** £18 / $21

The Raspberry Pi 3B+ is the first Pi model to include proper Power-over-Ethernet support, although you need one of these HATs to get it working. Of course, the network needs to be set up to support (i.e. power) the PoE and Raspberry Pi combo. It also currently only works on the Raspberry Pi 3B+.

**Use it with:** Server racks, headless solutions



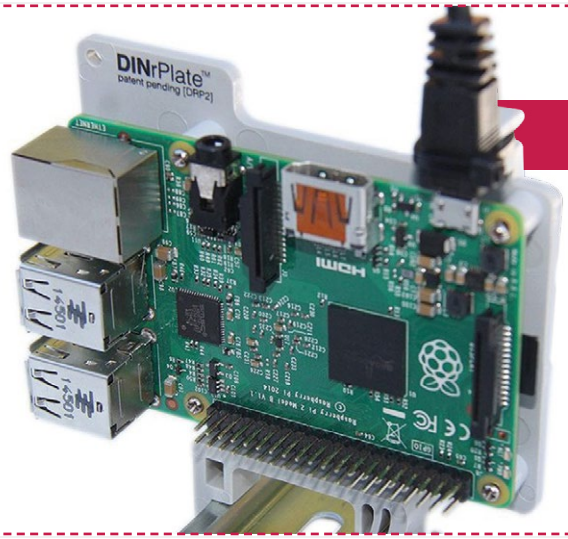

## Witty Pi 2

**magpi.cc/merPpf**

**price:** £20 / $26

Serious Raspberry Pi users often need power management and a real-time clock (RTC) for various reasons. The Witty Pi 2 offers these things, along with some extra on/off functions that can programmed. As the Raspberry Pi normally requires the internet to check the time, an RTC is essential for scheduling tasks on a Pi that's not online.

**Use it with:** Industrial settings, automated solutions




## DINrPlate DIN rail mount

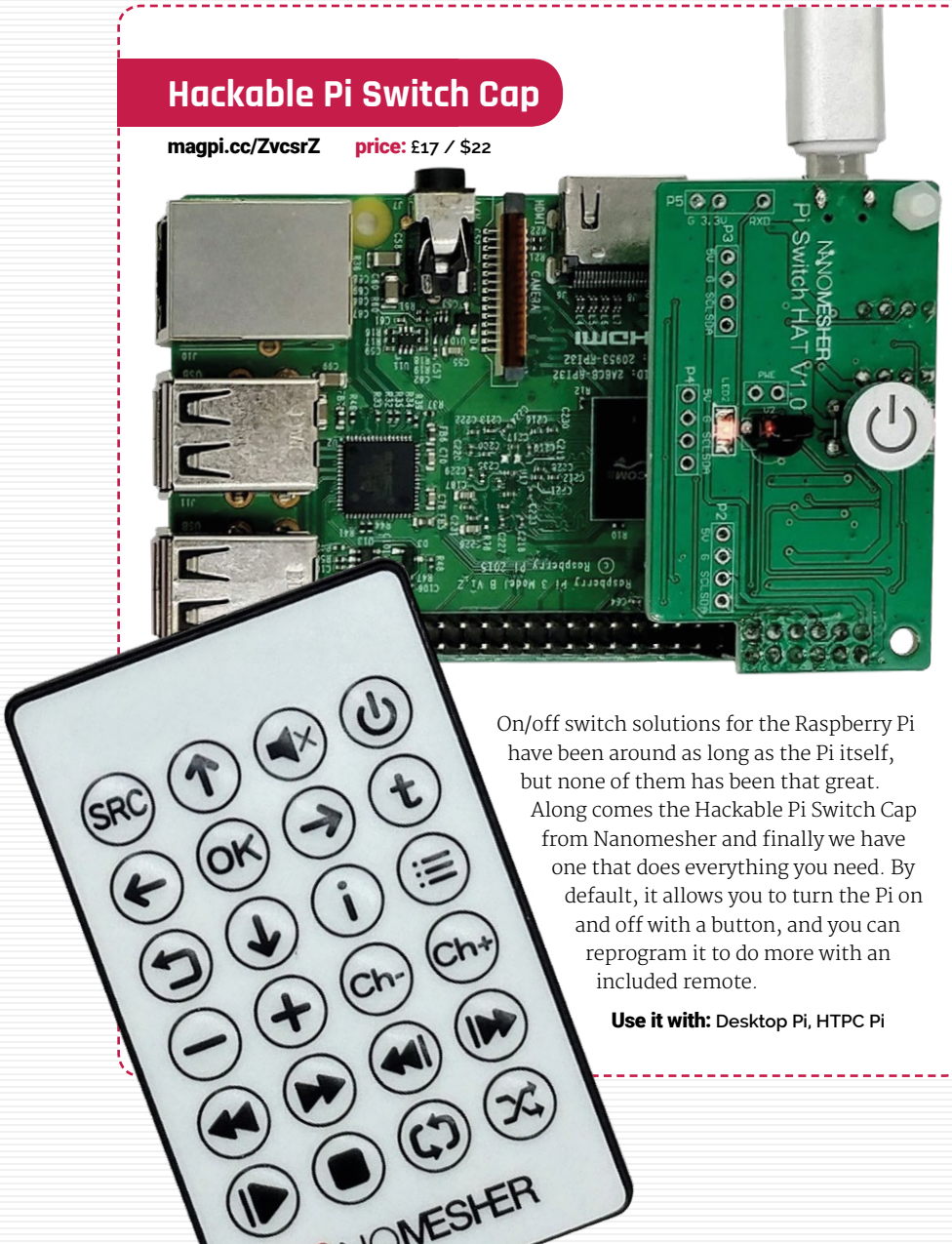
**dinrplate.com**
**price:** £10 / $13

Want to use a Raspberry Pi in a server environment, but don't need a UPS or RTC or any of the advanced stuff like that? Just have some DIN rails handy anyway? Then the DINrPlate is an excellent add-on that lets you mount a Pi to a DIN rail without any fancy extras. And the price reflects that.

**Use it with:** DIN rails, DIY servers

## Hackable Pi Switch Cap

**magpi.cc/ZvcsrZ** **price:** £17 / $22




On/off switch solutions for the Raspberry Pi have been around as long as the Pi itself, but none of them has been that great. Along comes the Hackable Pi Switch Cap from Nanomesher and finally we have one that does everything you need. By default, it allows you to turn the Pi on and off with a button, and you can reprogram it to do more with an included remote.

**Use it with:** Desktop Pi, HTPC Pi

## Strato Pi CM

**magpi.cc/rfBfvo**
**price:** £200 / $256

Are you a sysadmin in a big company that would like to use Raspberry Pi boards? The Strato Pi turns a Pi into a server good enough to pass the rigorous tests of enterprise IT. It has an impressive list of features: RTC, UPS, two-module DIN-rail case, RS-485, and more.

**Use it with:** Enterprise, servers

# MEDIA ADD-ONS

Improve your **listening, watching,** and **more**

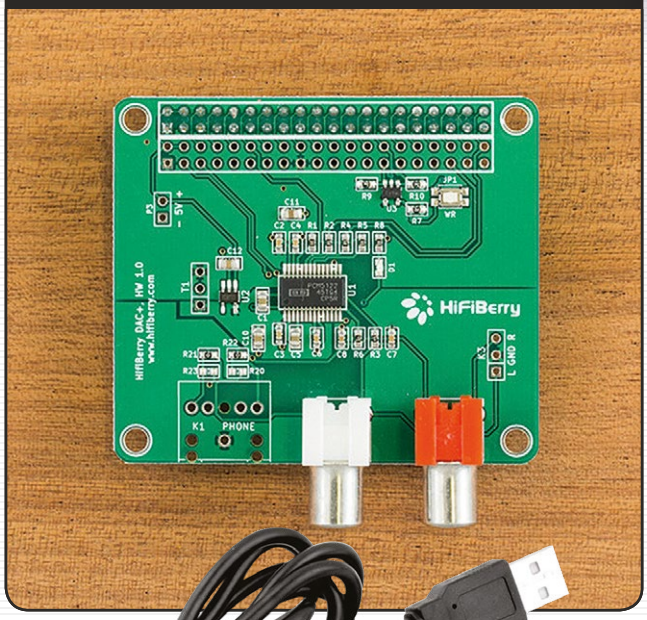
## HiFiBerry DAC+

**magpi.cc/aimAyy**

**price:** £23 / $29

There are many digital-to-analogue converters (DACs) that turn your Pi into a powerful stereo, but we're pretty partial to the HiFiBerry DAC+ range. There are cheaper versions, smaller versions, more powerful versions, and all of them let the Pi sound a bit better than normal.

**Use it with:** Stereo system, PA



## HyperPixel 4.0

**magpi.cc/WgSmAA**

**price:** £40 / $50

The HyperPixel 4.0 is a massive, wonderful screen for the Pi that you can get with or without a capacitive touch function. It's dead simple to set up and, due to the high pixel density, it's great for watching media or looking at pictures.

**Use it with:** Portable media projects, wall screens

## USB SNES controller

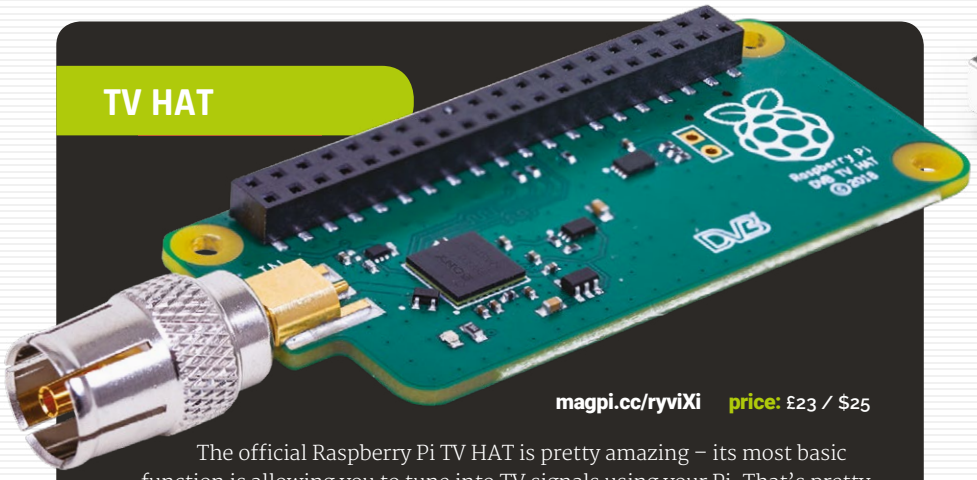
**magpi.cc/2ve6iUo**

**price:** £6 / $8

If you're playing retro games with your Raspberry Pi, you might want a retro controller to help it feel more authentic. We recommend one of the many versions of the classic SNES controller, with enough buttons to play any game prior to 1996 and some afterwards as well.

**Use it with:** Retro games console, RC car projects

## TV HAT

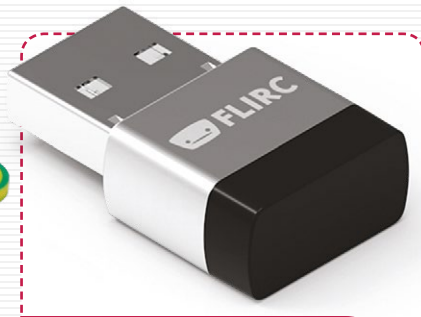


**magpi.cc/ryviXi** **price:** £23 / $25

The official Raspberry Pi TV HAT is pretty amazing – its most basic function is allowing you to tune into TV signals using your Pi. That's pretty standard, but the amazing part is its ability to stream said TV not only around your network, but also over the internet if you're abroad and missing the latest episode of *GBBO*.

**Use it with:** DIY DVR, TV anywhere

## FLIRC USB



**flirc.tv**

**price:** £20 / $23

Remote controls do have some good standards these days, and for stuff like Kodi there are plenty of ways to control your Pi-based HTPC via your phone or tablet. FLIRC makes the whole thing a bit simpler, though: take your favourite remote, let FLIRC know which button is which, and then just plug it into your HTPC. Simple.

**Use it with:** Kodi setups, remote-controlled IoT

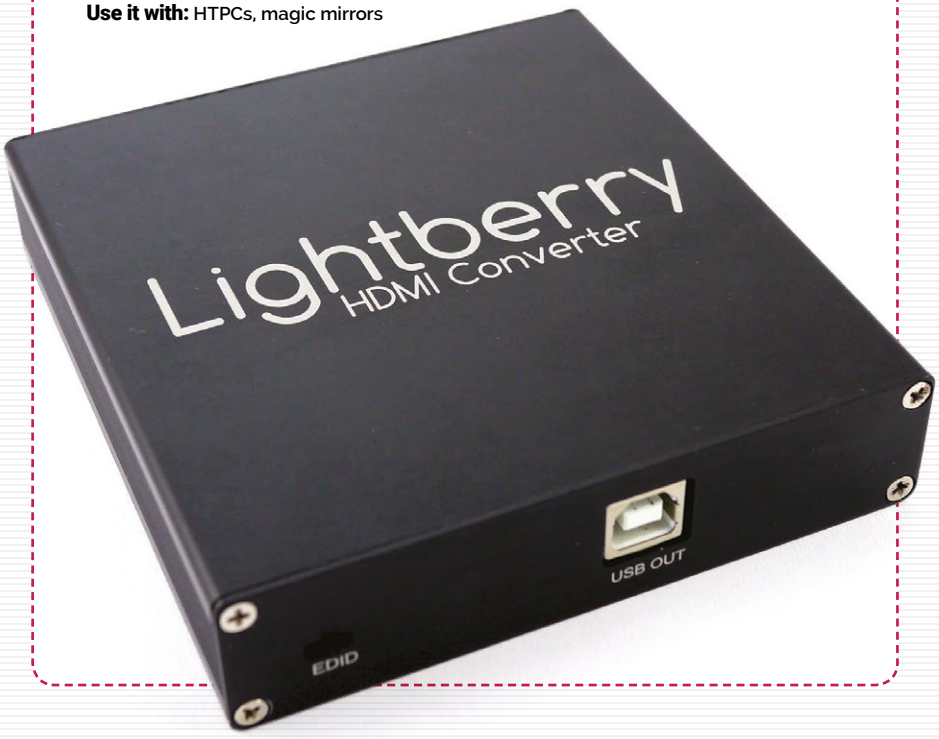
## Lightberry HD

**lightberry.eu**

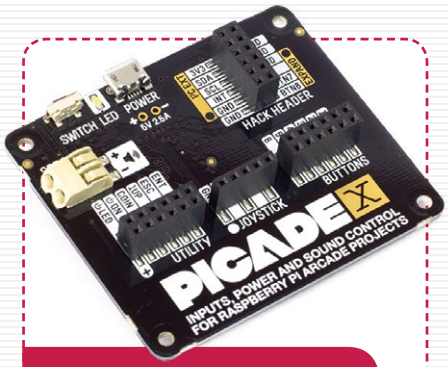
**price:** £95 / $120

Ever wanted your films to expand out of your TV? With Lightberry, you can install DIY Ambilight-like effects to your TV so that special effects become just that little bit more special. There are various kits to choose from as well.

**Use it with:** HTPCs, magic mirrors




## Picade X HAT




**magpi.cc/BupAFF**

**price:** £15

The Picade is Pimoroni's excellent tabletop arcade kit, and its newest iteration is controlled by the powerful Picade X HAT. It's so good, they made it so you could buy it separately to use in your own DIY Pi arcade setup.

**Use it with:** Arcade builds, remote controls

# PiDP-11

▶ Obsolescence Guaranteed ▶ **magpi.cc/wgWNTC** ▶ From $250

Turn a Raspberry Pi into a blinktastic classic 1970s computer? **PJ Evans** puts on his Paisley shirt and heats his soldering iron

**SPECS**

**DIMENSIONS:**
17×31×6cm

**MODEL:**
PDP-11/70

**ARCHITECTURE:**
16-bit

**OS:**
RSX-11M Plus

**BLINKENLIGHTS:**
64

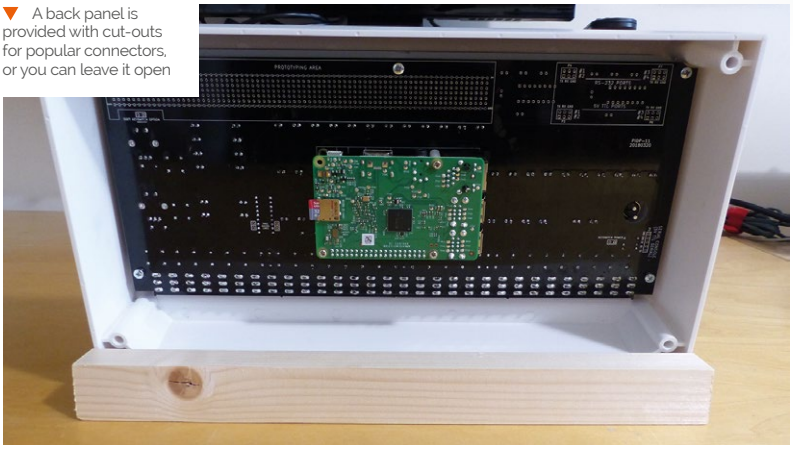
The launch of Digital's PDP-8 minicomputer in the 1960s was a defining moment in computing history, laying down the foundations of the hardware and software architectures we use today. Both it and the later PDP-11 were not only powerful machines, but also beautifully designed objects.

Oscar Vermeulen, an admirer of PDP range, has sold over 2,000 of his PiDP-8 replica: a Raspberry Pi-powered emulator with a fully functional one-third scale front-panel. Now comes his PiDP-11 kit. Released in 1970, the original PDP-11 is the most successful 'mini' computer in history, with over 600,000 sold.

## Remarkable replica

For this new kit, a painstaking process has resulted in an injection-moulded replica of the original PDP-11's case. If not for the one-third scale, you would struggle to tell it apart from the real thing. A perfect facia and custom-built switchgear complete the package. You even get a key and lock, just like the real thing.

▲ The completed PiDP-11 on the provided wooden stand



▼ A back panel is provided with cut-outs for popular connectors, or you can leave it open

Once built, the PiDP-11 PCB comprises 64 LEDs, two rotary encoders, and an array of switches that connect to the Pi's GPIO. Running a special version of the SimH emulator, the Pi accurately handles input and output from the panel. You can hook up a screen if you wish, use SSH, or go old-school and implement RS-232. The back panel is provided with different cut-outs to suit your cabling.

## Digital-it-yourself

The PiDP-11 is supplied in kit form and there's a lot to do. You'll need to have some experience in soldering to put this together, the focus being on accurately fitting the switches and LEDs. This is tricky, but Oscar has provided jigs that make the

> An essential purchase for anyone with an interest in computing history

alignment of all these components much easier than with the PiDP-8. The instructions are in an alpha stage, but they are clear and the switch section is especially detailed. It took us about five hours to complete.

Full instructions are provided on how to prepare the Pi for its new career in 1970s computing. At the time of publication, a one-stop SD card image should be available. Otherwise, there are a few hoops to jump through, but nothing too arcane and the steps are well explained.

Once you log in, you're straight into the PDP-11's operating system, an early form of UNIX. A number of alternative OSes are available, with more promised soon. You can switch back to Raspbian any time you like. In fact, as SimH doesn't put a lot of strain on the Pi, it is unlikely to struggle with other server tasks. As a result, many users have their PiDPs doubling up as file or media servers.

These kits are a labour of love for Oscar and the attention to detail shines through, from the quality of the casing to the extensive labelling on the PCB. You many find the price high, but the quality is there to match. An essential purchase for anyone with an interest in computing history.

## Verdict

The PiDP-11 ticks all the boxes. It's straightforward to build, beautifully cased, and is endlessly customisable. Whether you're interested in early computing or hypnotic flashing lights, you'll be delighted.

**9**/10

# Picade Console

▶ Pimoroni ▶ **magpi.cc/BSeTDD** ▶ £60 / $64

**SPECS**

**BOARD:**
Picade X HAT

**CONTROLS:**
Joystick, 6 × arcade buttons, 4 × utility buttons, illuminated power button

**SPEAKER:**
2.5-inch, 3W, 4Ω

**DIMENSIONS:**
245×120×140mm

The Picade's smaller sibling is an awesome compact console to plug into your TV. **Phil King** enjoys some more retro gaming

Following the arrival of a brand new version of the Picade – reviewed in *The MagPi* #74 (**magpi.cc/74**) – Pimoroni has given the same treatment to the Picade Console. Unlike its bigger brother, this retro gaming machine lacks its own screen, so you need to connect it up to a TV or monitor via HDMI. It does, however, pack an internal speaker so you'll still get sound even if your monitor doesn't have any. All you need to add is an HDMI cable, microSD card (with the RetroPie OS on it), and a Raspberry Pi – any 40-pin model will work, but we'd recommend a Pi 3B+ for emulating some of the more powerful retro systems.

## Putting it together

The lack of a screen does make the Picade Console a whole lot easier to assemble than the full-size Picade. Made up of five black powder-coated MDF panels (with helpful labels), its 'cabinet' is essentially identical to the Picade's control console section, but with an extra rear cut-out for the Pi's HDMI port. Assembly instructions are on the

▶ A compact console, it packs all the features of the full-size Picade apart from the screen

▲ You can plug it into any TV or monitor with an HDMI port

back of an A3 poster, or you can use the online ones (**magpi.cc/wxbMLu**).

It took us around an hour to put together. The panels are connected using plastic brackets and metal M3 nuts and bolts. The only tricky part was reaching in to place the tiny nuts on the side bolts (tip: put a bit of Blu Tack on your finger). For the top section, the supplied artwork (or your own) is sandwiched between the black panel and a clear Perspex one. A microswitched joystick (with ball top) and push-fit arcade buttons are then inserted – or you could customise it with your own, such as Pimoroni's Plasma buttons (see boxout).

## PLASMA BUTTONS

To add some extra razzle-dazzle to your Picade Console (or Picade), you could swap out the standard arcade buttons for Pimoroni's illuminated Plasma buttons (£35 / $37). Each clear button features a mini PCB with four tiny RGB LEDs on it. Daisy-chained together, the PCBs are connected to the Picade X HAT's Hacker header for power and control – you can program dynamic lighting effects and patterns.

▲ All the parts of the kit – assembly is fairly straightforward

Once the mini speaker is fitted, you can mount a Raspberry Pi on the base and add the key part of the system: the new Picade X HAT. Also available separately (£15 / $16) for those who want to build their own custom arcade machine, the HAT has easy-to-use DuPont connectors for the numerous joystick and button wires. The only slight issue with the Picade Console – which is more compact than its original incarnation – is that there's not a lot of room inside for all the wiring.

The Picade X HAT also features a built-in I²S DAC and amplifier for the internal speaker, and power management that allows you to safely shut down the Raspberry Pi with the illuminated power button on the side of the cabinet.

### Simple setup

With everything assembled, you just need to plug in a keyboard for the simple software setup: installing the Picade X HAT driver with a one-line Terminal command. With RetroPie running, it's a case of setting up the joystick directions and buttons to your liking and – once you've added some ROM files – you're ready to play your favourite retro games.

As with the full-size Picade, the unit feels robust, with rubber feet to keep it secure on a desk or table. The arcade controls are solid, although the buttons are leaf-spring rather than microswitched. One slight drawback of the Picade Console's smaller form factor is that the side utility buttons are nearer the front and on occasion we knocked them accidentally while playing. Other than that, it's just as impressive as the Picade, but considerably less expensive.

## Verdict

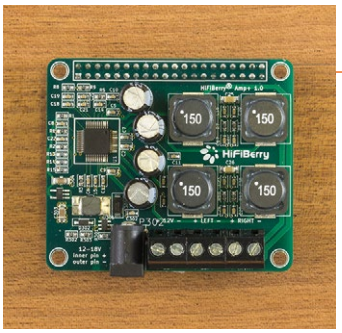
With a robust case and controls, and smart Picade X HAT board, it provides an arcade-style experience on your TV screen at a very attractive price.

**9**/10

# 10 Best: DACs

Improve the sound quality of your Raspberry Pi with these digital-to-analogue converters

Let's not put the Raspberry Pi down here – it's not like its audio output sounds like a worn-out video tape in 1992. You can get some good sounds from it as standard – however, it can still be made much better. This is where a whole host of amazing DAC accessories come in.



## HiFiBerry Amp+

Going beyond a DAC is this amplifier that you can use as part of a serious stereo system. It's Class D and all you need to do is connect your loudspeakers. It's great for multi-room setups.

- £45 / $57
- **magpi.cc/KWNSpN**

## Nanosound DAC

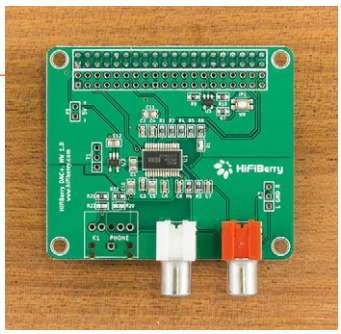
This powerful DAC comes in a variety of versions for different audio tastes, and also includes physical buttons, a remote, and you can even get a special case for it. A couple of models even come with a little screen to see what you're listening to!

- From £38 / $48
- **magpi.cc/XdxPxP**

## HiFiBerry DAC+

The HiFiBerry DACs are pretty popular, and come in a range of sizes and different SKUs for any budget, audio requirements, or Raspberry Pi type. There's also a digital/SPDIF version.

- From £20 / $26
- **magpi.cc/aimAyy**



## Nanosound Player

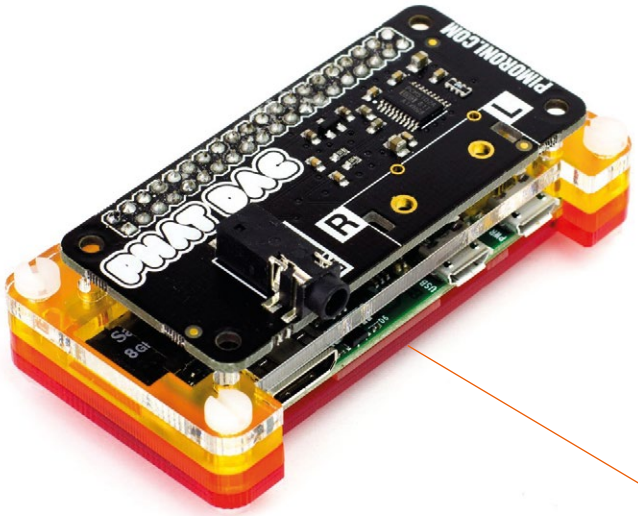
This all-in-one kit uses the Nanosound AMP to create a complete stereo system to which you just need to connect some speakers. It also comes with the remote control of the other DACs and AMPs, and it's all in one neat case.

- £132 / $169
- **magpi.cc/tCRnde**

## Pi-DigiAMP+

IQAudIO is one of the original brands producing audio add-ons for the Raspberry Pi, and its DigiAMP range has recently been updated while still being an excellent product.

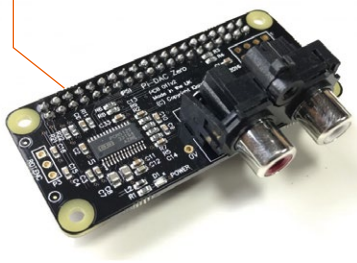
- £54 / $69
- **magpi.cc/fisaLH**




## Pi-DACZero

This Zero-sized DAC is the sibling to the full-sized audio DACs for other Pi models. Considering the standard Pi Zero doesn't have an audio out port, you'll need something like this for almost any solution where you want the Pi Zero to play sound.

- £17 / $21
- **magpi.cc/sxVfhU**



## Speaker pHAT

This DAC goes beyond being able to output music to a stereo – it makes the Pi Zero a stereo thanks to its tiny built-in speaker. It even has a volume graph to show you just how loud your sick tunes are.

- £12 / $13
- **magpi.cc/2kXdZsE**

## pHAT DAC

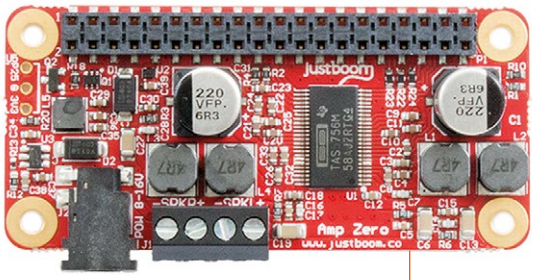
This nice and inexpensive DAC adds a 3.5 mm audio out port – i.e. a headphone jack – to a Pi Zero. Which is neat, as it doesn't normally have one. You can also solder RCA ports on to expand it.

- £12 / $13
- **magpi.cc/SUsJrP**

## Digi HAT

With optical and coaxial out, you'll be able to get some super high-quality music with the Digi HAT. JustBoom also sells traditional DAC versions with standard RCA sockets, if you prefer that.

- £30 / £39
- **magpi.cc/2lMEYFx**




## Amp Zero pHAT

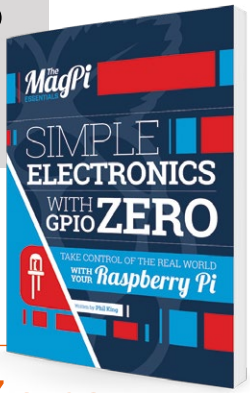
An amp for a Pi Zero? Absolutely, and this is one of the few we've found. JustBoom produces a great range of amps and DACs and this is no exception – it's just a lot smaller.

- £24 / $37
- **magpi.cc/2iNJMKP**

## MUSIC PLAYERS

With the latest version of Raspbian, VLC is now the default media player on the Raspberry Pi. It handles many kinds of codecs and is highly configurable, making it the perfect software for your DAC-powered Pi.

# Learn basic electronics with Raspberry Pi




**Gareth Halfacree** takes a look at the best resources for those looking to break in to the world of electronics

## Simple Electronics with GPIO Zero

AUTHOR

**Phil King**

Price:
£4 / $5 (free download)
magpi.cc/gpio-zero

Written by *The MagPi*'s contributing editor Phil King, with assistance from contributors Mike Cook, Richard Hayler, and Ben Nuttall, *Simple Electronics with GPIO Zero* is a one-stop guide to getting started with Raspberry Pi-controlled electronics.

Inside, you'll be taken on a journey from learning about individual components and reading circuit diagrams to building real-world projects including programmable lighting, a quick-fire reaction game, an intruder alarm, and even a working robot.

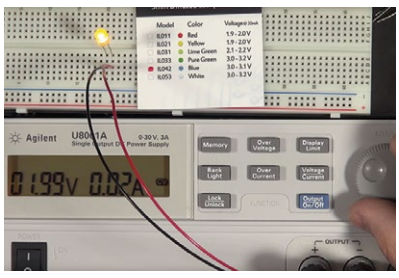
All the projects are built around GPIO Zero, an easy-to-use Python library for controlling the Raspberry Pi's general-purpose input/output (GPIO) header. A reference guide at the back walks through all the functions one-by-one, from controlling LEDs to spinning motors, making it a handy thing to have around even when you've worked through all the projects.

All the projects in the book are illustrated with easy-to-read diagrams and full-colour photographs, making it simple to match your build to the instructions.

Like all *The MagPi* Essentials books, *Simple Electronics with GPIO Zero* is available to buy in print and digitally, or as a free download under the Creative Commons licence.

## Concepts Videos

Get a handle on what electricity actually is



### PRACTICAL NINJAS' BASICS OF ELECTRICITY AND ELECTRONICS

If you want to take a step back from practical projects and understand exactly what electricity is and how it works, these short videos are a great starting point.
magpi.cc/LeyJpu

### BEN EATER'S DIGITAL ELECTRONICS TUTORIAL

This ten-part video series dives into the inner workings of components including LEDs and transistors, using hands-on investigation to explain the mechanics of what's happening in the simplest of circuits.
magpi.cc/kqiMNF

### ALL ABOUT CIRCUITS' VIDEO LECTURES

Aimed at a college-level audience, Tim Fiegenbaum's lecture series covers a wealth of topics, from basic electronic concepts all the way through to complex devices like microprocessors and modern computers.
magpi.cc/DqrnAT

# CamJam EduKits

AUTHOR
**Cambridge Raspberry Jam**

Price:
£4 / $5 (free download)
magpi.cc/pEKWSQ

Created by volunteer organisers of the Cambridge Raspberry Jam in collaboration with The Pi Hut, the CamJam EduKits combine all the components you need with easy-to-follow worksheets suitable for home or classroom use – meaning you don't need to worry about buying the right components separately.

There are three kits available: the Starter kit, which includes the components and instructions needed to build simple projects with buttons, LEDs, and a buzzer; Sensors kit, which adds temperature, motion, and light sensing components; and the largest, Robotics kit, which has the components needed to build a wheeled robot.

The kits are designed for use with Python via GPIO Zero, though community-provided Scratch programs are also available for most projects. All worksheets are free to download under a Creative Commons licence.

# Raspberry Pi Workshop for Beginners

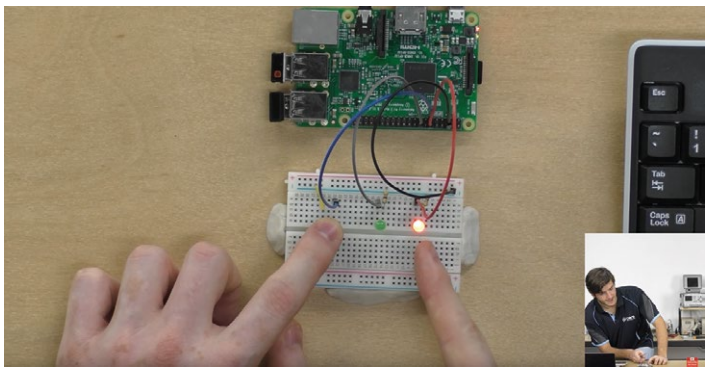
AUTHOR
**Core Electronics**

Price:
Free
magpi.cc/YCOsZj

If you're new to Raspberry Pi itself, rather than just electronics, Core Electronics' beginners' workshop course is a great introduction. Through a series of hands-on videos, coupled with a webpage for circuit diagrams and code extracts, you'll learn everything from how to load an operating system onto your Pi to using its GPIO header to drive a simple circuit.

Core Electronics' course goes a little further than most: once you're comfortable with the basics, you can continue watching the video series to learn more advanced topics like shell scripting, writing graphical user interface (GUI) programs, and even connecting your Pi up to the Internet of Things (IoT).

All the videos are filmed in high quality and are free to watch via the company's YouTube channel.



## Electronics Kits

Get everything you need in once place

### MODMYPI YOUTUBE WORKSHOP KIT

Designed for use alongside an accompanying nine-part video series with freely downloadable worksheets, ModMyPi's bring-your-own-Pi kit covers a range of basic projects in Python using the RPi.GPIO library.

- magpi.cc/apwuju
- £16 / $20

### MONK MAKES STARTER KIT

Put together by noted technology author Simon Monk, the Monk Makes Starter Kit comes with a basic selection of components, plus printed instruction cards with diagrams for ten interesting projects.

- magpi.cc/fiuyya
- £13 / $16

### VELLEMAN RASPBERRY PI BASIC LEARNING KIT

Designed for the more advanced user, Velleman's kit comprises 75 pieces including a handy GPIO extension board, remote control, servo motor, flame sensor, and seven-segment LED displays.

- magpi.cc/RqPYWD
- £63 / $65

# Alex Martinez

A desire to compete in robotics brought workshops in Raspberry Pi to a Puerto Rican school

> Category Educator | > Day job Teacher | > Website magpi.cc/SBynYZ

"My background is really related to rock and soils," Alex mentions to us. He's recently tagged *The MagPi* Twitter account into a post about his workshops in Puerto Rico. "I graduated from the Department of Geology of the University of Puerto Rico. I took a C++ course which I found very interesting and that was it, until the maker movement hit me five years ago."

Alex helped start a makerspace with his friends and family in 2014, and he learnt all about microcontrollers and the Raspberry Pi from the community that gathered there.

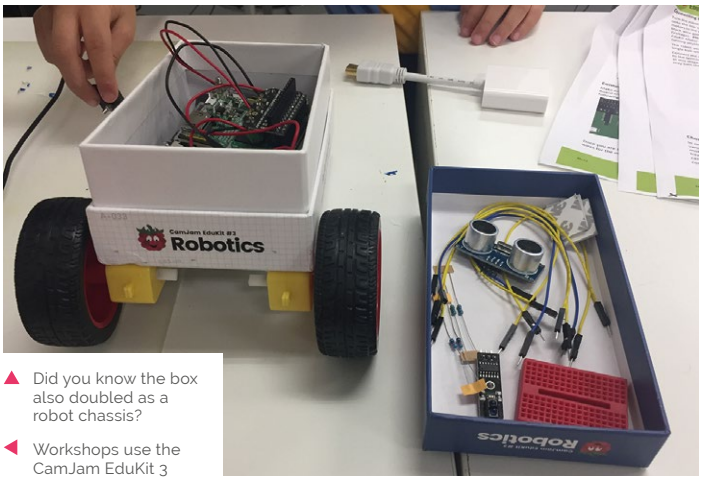
"Then, back in 2016 I went to US to be part of the first cohort of Raspberry Pi Certified Educators," Alex reveals. "Since then I have been introducing the Raspberry Pi ecosystem to the classroom and helping fellow educators in the island with talks and workshops along with fellow makers."

Like us, you may be wondering how the US territory of Puerto Rico is doing after Hurricane Maria devastated it last year. Alex himself refers to Puerto Rico today as paradise, but it's clear there are still some long-lasting effects:

"The hurricane hit us very bad. We had to cancel many maker-related events and gatherings. Many of our friends still work on their personal projects on their houses, but we lost most of the momentum by staying closer to family."

**What prompted you to start the workshops?**

The current workshop that we are giving in my school started because my school Director wanted me to enter a robotic league competition, but the fees and equipment were very expensive and we couldn't




▲ Did you know the box also doubled as a robot chassis?

◀ Workshops use the CamJam EduKit 3

- Students learn about code
- People from around the island attend the workshops
- Making robots is fun, but sometimes hard work
- Pi Wars inspired Alex's workshops

afford it. Most of the educators I talked to are experienced in VEX [robotics] competitions and had been in the US finals, but never had an experience on any Raspberry Pi-related projects. This was another reason to start these workshops: many of them wanted to build projects from scratch and teach their kids computer programming skills. The Raspberry Pi community is very resourceful and here we found the Pi Wars events. So I sent a message to Pi Wars organisers in order to ask for permission to make a Pi Wars-like event with fellow educators here in Puerto Rico. I asked my colleague Damaso Cardenales – who is a member of our Makerspace PR, Inc., and a computer programmer – to help me with this project.

> Many of them wanted to build projects from scratch and teach their kids computer programming skills

**What kind of things do you teach in the workshops?**

For these workshops we are using the CamJam Robotics kits that you can find in The Pi Hut.

The majority of the teachers and students are new to the platform, so the workshops are divided on the level of difficulty. Teaching [ranges] from how to burn a Raspbian image on the SD card, to working with the GPIOs using Python, and remotely accessing Raspbian from another computer. We have been giving three workshops until now and are planning more in the future.

**How well have the workshops been received?**

In my opinion, the workshops were so good that we had home-schoolers coming in, and parents with kids attending the sessions. There is a great effort from other schools to bring these workshops closer to them. Also, we are planning to hold a Pi Wars event next year.

For next year, I am planning – with the awesome people of MakertechPR, a local DIY and electronics store – a Raspberry Pi summer workshop for the University of Puerto Rico.

## Helping out the Puerto Rican maker community

If you're interested in getting involved with the Puerto Rican maker community, Alex is happy to talk to you. You can contact him at:

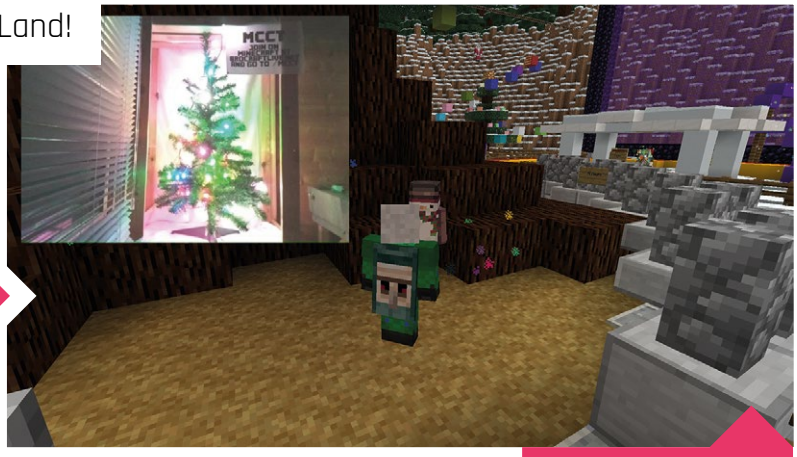
- Twitter @geomarster
- Facebook facebook.com/groups/makerspacepr
- Email geomarster@gmail.com

# This Month in **Raspberry Pi**

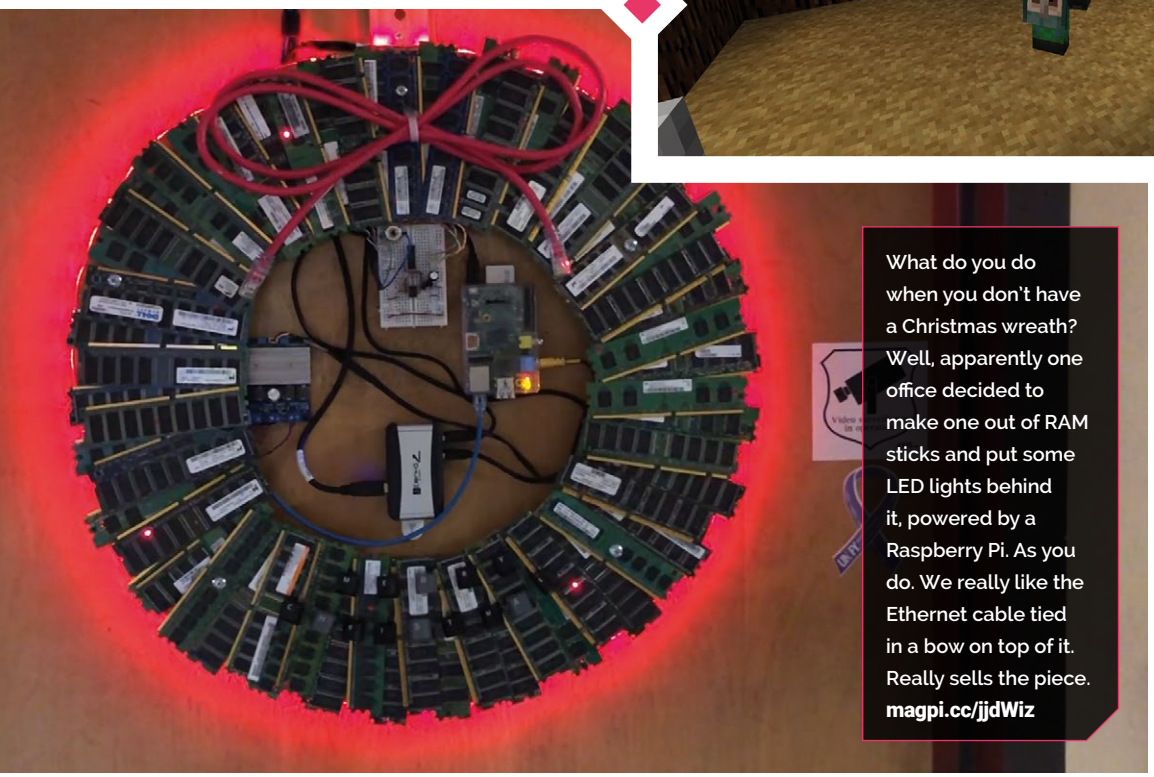
# Christmas **2018**

Season's greetings from Raspberry Pi Land!

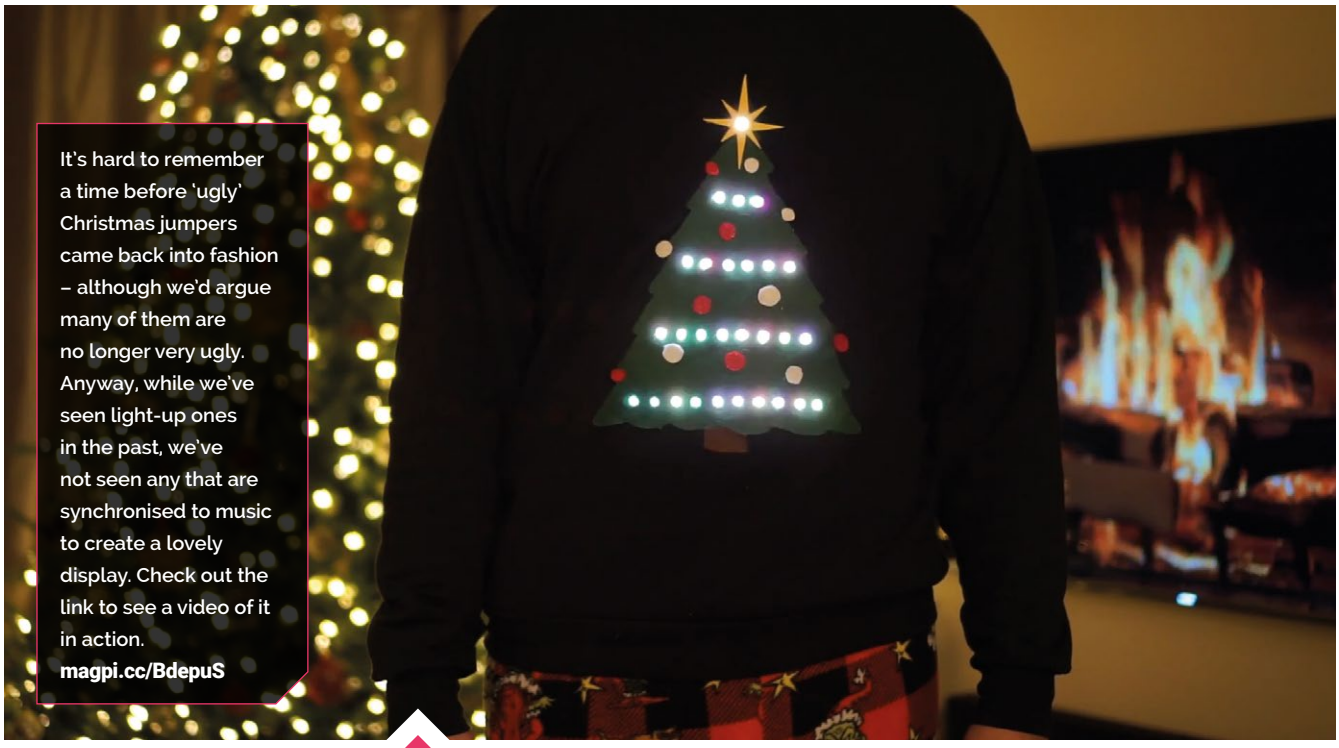
Although this is our January issue, you very likely might be reading this right before Christmas. So to fleetingly keep in theme with the season, here are some of the great Christmas Pi projects we've seen this month!



Using Minecraft Pi, the folks at Brocraft Gaming are controlling a real-life Christmas tree. You can interact with it by jumping into their server on Minecraft: Java Edition for PC and Mac, gathering power, and talking to an NPC below an in-game tree to change the light configuration on both trees. Make sure you have the live stream on so you can see the changes you make!
**magpi.cc/VBxjcQ**



What do you do when you don't have a Christmas wreath? Well, apparently one office decided to make one out of RAM sticks and put some LED lights behind it, powered by a Raspberry Pi. As you do. We really like the Ethernet cable tied in a bow on top of it. Really sells the piece.
**magpi.cc/jjdWiz**



It's hard to remember a time before 'ugly' Christmas jumpers came back into fashion – although we'd argue many of them are no longer very ugly. Anyway, while we've seen light-up ones in the past, we've not seen any that are synchronised to music to create a lovely display. Check out the link to see a video of it in action. **magpi.cc/BdepuS**

This massive array of lights is controlled by a Raspberry Pi and two microcontrollers. It's constantly shifting and changing colours, so it's pretty magnificent to look at in motion – check out a video in the link provided! We have some Pi-powered Christmas lights, but nothing as ridiculous as this. **magpi.cc/TdhEFr**

This office decoration plays random Christmas and Holiday commercials, which you can shuffle at the press of a button. Apparently it also plays trailers for Christmas classics such as *The Muppet Christmas Carol*. We hope it has trailers for other favourites like *Lethal Weapon*, *Kiss Kiss Bang Bang*, *Iron Man 3*, and other movies from the filmography of the King of Christmas himself, Shane Black. **magpi.cc/DzENuQ**

# New UK Scouts resources!

Find the Scouts resources here! **magpi.cc/nwriRe**

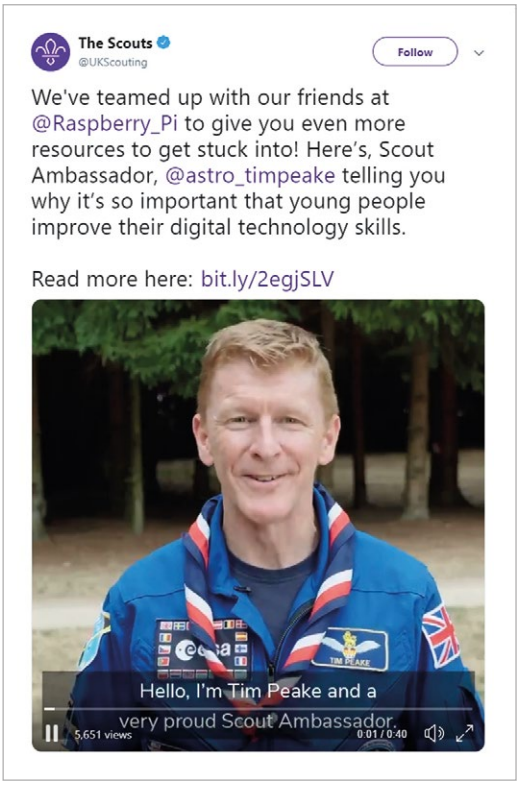
The Digital Maker badge gets upgraded

Six months ago, the Raspberry Pi Foundation announced a partnership with the Scout Association in the UK, launching the Digital Maker Staged Activity Badge. As you can see from the number on the badge, they're 'staged' to several levels. Resources were released for stages 1 and 2 and now the Foundation is releasing more, as Olympia Brown, Senior Programme Manager explains in her blog post:

"Since then, we've been developing resources for more stages of the badge, and we've just released activities to support more of stage 2 and stage 3.

> The first set of activity resources we released either needed no technology or laptops only

"Because the Digital Maker badge is a staged activity badge, any section of the Scouts movement can tackle it. And since an activity that interests and engages a Beaver is likely to be quite different to one that engages an Explorer Scout, we've increased the variety of activities we're providing.


The Scouts
@UKScouting
Follow
We've teamed up with our friends at @Raspberry_Pi to give you even more resources to get stuck into! Here's, Scout Ambassador, @astro_timpeake telling you why it's so important that young people improve their digital technology skills.
Read more here: bit.ly/2egjSLV
Hello, I'm Tim Peake and a very proud Scout Ambassador.
5,651 views
0:01 / 0:40

▲ Tim Peake, the first Astro Pi astronaut, has helped promote the new badge!

## More tech!

"The first set of activity resources we released either needed no technology or laptops only, as the leaders we spoke to told us it shouldn't be too difficult to get hold of some laptops for a session. For the new resources, we've increased the variety of tech that we recommend using. Some of the activities use the micro:bit, since it's a low-cost, easy-to-use bit of tech. For leaders unfamiliar with the micro:bit, we've put together this guide on using the device: **magpi.cc/nyRqJj**.

## More activities!

"With all our activity resources, we show how digital making fits into the scouting movement and into many typical activities you'd do with your troop. For example, you can program the micro:bit to be the musical accompaniment to your next campfire (**magpi.cc/AdiJyc**). Or, you can create your own custom map to show points on a recent hike that you did together (**magpi.cc/GEzWYa**) – anything from where someone fell over, to where you saw the most amazing view."

# Crowdfund **this!**

Raspberry Pi projects you can crowdfund this month

## TinyPi Pro

We covered the original version of this project in *The MagPi* a while ago, and the creator has finally deemed it worthy to be made into a product. He needs your help, though, to raise enough cash to start production, and of course you can net yourself a kit for your pledge.

▶ **kck.st/2PdLFNV**

## Arcard

Tabletop arcade machines are great, but some of them can be a bit expensive, The Arcard brings down the price slightly by having a case made entirely of cardboard. It's also big enough for two sets of controls, which not every Pi-powered arcade can do

▶ **kck.st/2Ubmq2r**

**CROWDFUNDING A PI PROJECT?**

If you've launched an irresistible Pi-related project, let us know!

**magpi@raspberrypi.org**

# Best of the rest!

Here are some other great things we saw this month

### STEAM LINK ON RASPBERRY PI!

So this is pretty amazing news – although Valve has discontinued its Steam Link hardware, it's just released an app for Raspbian that does the same thing. We'll have more thorough coverage of it next issue, but for now… wow.

▶ **magpi.cc/pUFkTf**

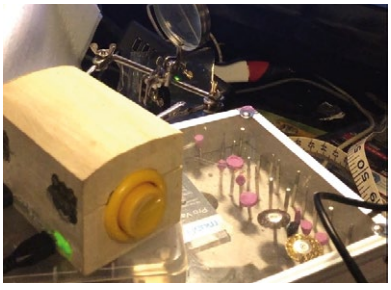
### CUSTOM VINTAGE PI COMPUTER

Taking an old Beckman terminal, Imgur user 'iafan' has converted it to use a Pi. The old CRT was replaced with an LCD screen, and the remaining space was used to house the Pi and other components. It looks pretty cool.

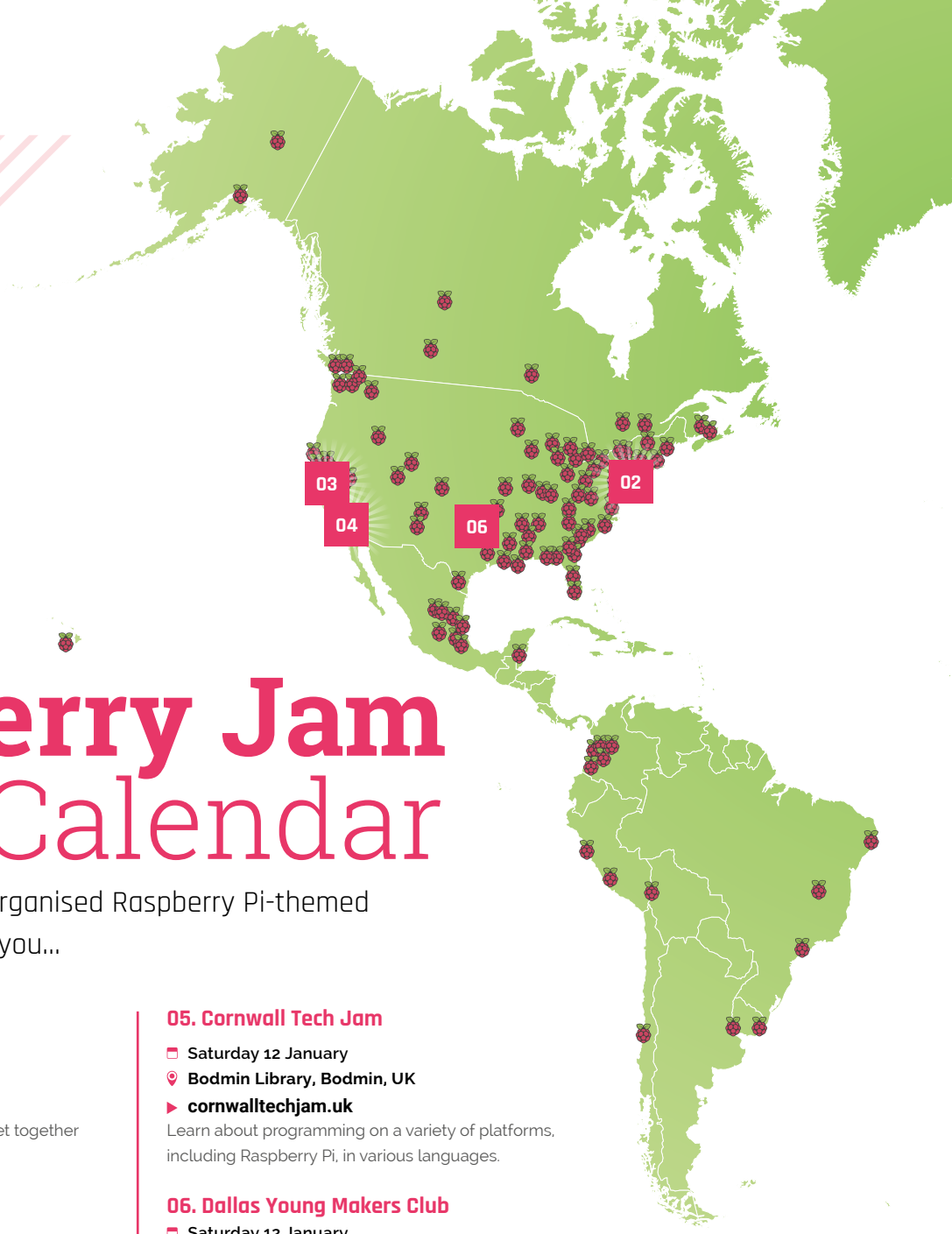
▶ **magpi.cc/sLDRcG**

### FRASIER SHUFFLER

*The Simpsons* shuffler was great, but we also love this *Frasier* shuffler. The wooden case and big arcade button give it a nice look, and you could probably make the case resemble the cask for a 1982 Château Latour to suit Frasier's tastes.



▶ **magpi.cc/HctpPd**




# Raspberry Jam Event Calendar

Find out what community-organised Raspberry Pi-themed events are happening near you...

### 01. Stafford Raspberry Jam

- Tuesday 8 January
- **Stafford Library, Stafford, UK**
- **magpi.cc/PESDWA**

A big meet-up for people of all ages to get together share their Pi ideas while having fun.

### 02. DC Raspberry Jam

- Saturday 12 January
- **Mount-Pleasant Library, Washington DC, USA**
- **magpi.cc/nmxWjr**

An event for kids to get into Pi-based robotics and coding. There's also a 3D-printed robot to have fun with.

### 03. Raspberry Jam @ Castro Valley Library

- Saturday 26 January
- **Castro Valley Library, Castro Valley, CA, USA**
- **magpi.cc/xjipLg**

If you're interested in coding and want to know more about the Raspberry Pi, check out this Jam.

### 04. Saddleback Valley Raspberry Jam

- Saturday 2 February
- **Laguna Hills High School, Laguna Hills, CA, USA**
- **magpi.cc/kwDpdp**

Projects will be on display from community members, and there will also be beginners' workshops.

### 05. Cornwall Tech Jam

- Saturday 12 January
- **Bodmin Library, Bodmin, UK**
- **cornwalltechjam.uk**

Learn about programming on a variety of platforms, including Raspberry Pi, in various languages.

### 06. Dallas Young Makers Club

- Saturday 12 January
- **J. Erik Jonsson Central Library, Dallas, TX, USA**
- **dallasyoungmakers.org**

Learn about computing and robotics on Raspberry Pi at this free workshop for kids!

### 07. January Raspberry Pint

- Tuesday 29 January
- **CodeNode, London, UK**
- **magpi.cc/PXtDiX**

If you're working on a Pi project, or would like some help on it, head down to Raspberry Pint!

### 08. Leeds Raspberry Jam

- Wednesday 6 February
- **Dixons Unity Academy, Leeds, UK**
- **magpi.cc/cqDJms**

Get hands-on with digital making activities in the workshop, and a hackspace area to share projects.

**FULL CALENDAR**

Get a full list of upcoming events for January and beyond here:

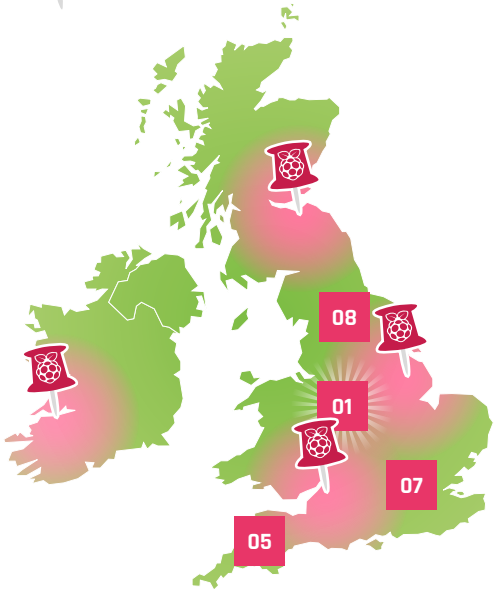
**rpf.io/jam**

## FIND OUT ABOUT JAMS

Want a Raspberry Jam in your area? Want to start one? Email Ben Nuttall about it:

**jam@raspberrypi.org**

We've highlighted some of the areas in need of a Jam! Can you help out?




## Raspberry Jam advice:

# Themed activities

"When a competition like Astro Pi, or a programme like Pioneers comes out, we'll do talks and invite people along to take part in workshops at the Jam. It's good because it gives an incentive for people to come along and take part, and gives them motivation to work."

**Andy Melder – Southend Raspberry Jam**

Every Raspberry Jam is entitled to apply for a Jam starter kit, which includes magazine issues, printed worksheets, stickers, flyers and more. Get the book here: **magpi.cc/2q9DHfQ**

# Your Letters

▲ These smart lights helped inspire 'oldjake' - did they inspire you?

## Steam Pi

**I'm really excited about Steam coming to the Raspberry Pi! How long has this been in the works? Will it be able to play all of the games in the Steam library? Do I need the latest Raspberry Pi for it? I have many questions! Can't wait for a special Steam Pi to replace the current Steam Link hardware!**

**Tyler** via Facebook

It's not actually a full Steam client on the Raspberry Pi – it's Steam Link, the streaming service. You'll need a computer already running Steam on your network to get it working, but you do get access to all the games available on your gaming PC! It was news to us as well, though – Valve is quite secretive.

You will need a Raspberry Pi 3 (Model B): either an original Pi 3 or a 3B+. We've not tested it on a 3A+, but we can only imagine that reduced RAM isn't ideal for it. Otherwise, just make sure you have as much of the network wired up as possible, and then just plug in a compatible controller! Check out the details here: **magpi.cc/pUFkTf**.

▲ With the old Steam Link hardware being discontinued, we were wondering what Valve might do!

## Let there be light

**If there was nothing else interesting in any other edition of *The MagPi*, I would still pay for them all just for the article about IKEA lights from last issue!**

**I've got loads of their bulbs around the house as I'm swapping all my lights from Hive to Trådfri. One of the things missing from the Trådfri platform is the ability to turn a light on at sunset – you can only set an absolute time.**

**So, this morning, I thought I'd have a play (I'm very new to Python so this isn't intuitive). Couple of hours later and I've got a little Python script that gets the sunset time, compares it with the current time, and switches the lights on if it's later than sunset. I'm running some other home automation stuff with a script that runs every few minutes, so it'll be dead easy to add this in.**

**For the record, as it's in the spirit of 'hacking' home automation, my pet project, which is what got me started in this, is occupancy detection so that I can control the heating depending on who's in. I've now got that 99% working. There's a Pi in the lounge that sniffs for the Bluetooth MAC addresses of the phones in the house. If it finds one, it increments a score. If the score is zero then it turns the heating down. I've even got it weighted so that if I'm in the house, it turns the heating up a bit more. I'm using the very badly documented Hive API to do this, so it's been a bit of a labour of love. When I started this, I knew nothing about Python or APIs, or much else that would help with this. I don't do 'Hello World!' so this has definitely been a bit of a baptism of fire for me. I've done a bit of a write-up here: magpi.cc/iHSdpB.**

**oldjake** from the forum

Wow! We're extremely impressed with the work you've done here. We don't want to toot our own horns too much but when we see readers taking a tutorial we've done and using it in a current project, it makes it all seem worth it!

For folks who are interested in how oldjake has fared so far, you can read more in the forum thread about his accomplishments here: **magpi.cc/uCrxJW**.

## The MagPiTV

**After reading the article entitled 'BBC Computer Literacy Archive' [from issue 73], it made me think...**

**What about a Raspberry Pi BBC programme? After watching the BBC Micro programme many years ago, eagerly awaiting each episode, I went on to eventually work in IT and now at 60 I have still not lost the enthusiasm I have for computers thanks to the BBC.**

**This made me wonder why the BBC has not produced a similar programme, brought up to date of course, using a Raspberry Pi instead of the BBC Micro, with maybe a new hardware project each week including how to create a program in Python to control it?**

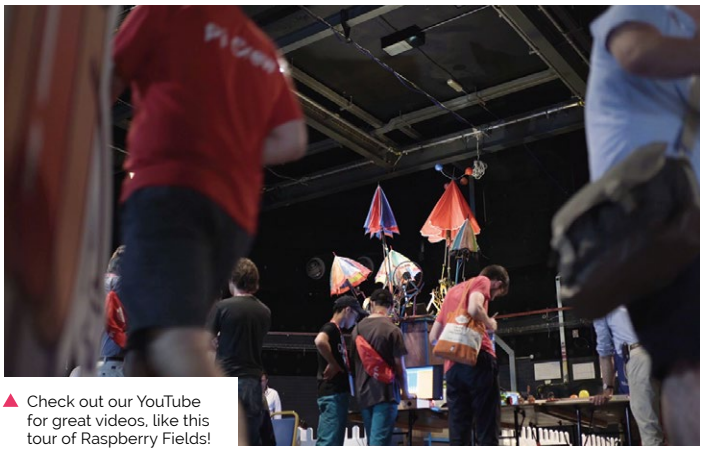
**I think any children with an interest in computers and the Raspberry Pi (which is in most schools) would love it, with the added bonus of again helping children in school with maths and logic problem solving. The low price of the Raspberry Pi makes it an ideal platform.**

**The Government seems supportive. An easy cheap programme to make? What do you think?**

**Mike** via email

If you've ever met our Features Ed Rob, or seen any of the videos on our YouTube channel (or even been to certain UK comic cons), you'll know he loves being in front of the camera. Unfortunately, it's up to the BBC to fund and develop that kind of programming.

Speaking of our YouTube channel, if you want to see Raspberry Pi content on a semi-regular basis, make sure to subscribe. We do unboxings, tutorials, walking tours, and more. Check it out at **magpi.cc/youtube**.



▲ Check out our YouTube for great videos, like this tour of Raspberry Fields!

## Contact us!

- Twitter **@TheMagPi**
- Facebook **magpi.cc/facebook**
- Email **magpi@raspberrypi.org**
- Online **raspberrypi.org/forums**

COOLEST PROJECTS 2019

Coolest Projects is a world-leading annual showcase that empowers the next generation of digital innovators: young people from across the whole Raspberry Pi community come together to exhibit their awesome projects.

Join Coolest Projects at The Sharp Project, Manchester on Saturday 2 March, and see them share their ideas and their passion for technology, make new friends, and learn from one another. Bring family and friends, see crazy science shows by TV presenters, and have a go yourself at the exciting activities.

Why not get involved yourself? You don't need to know how to code. They need volunteers with enthusiasm and a variety of skills for Coolest Projects. If you do know how to code, that's great, but you don't need any technical skills to help make this day an amazing event for young people.

coolestprojects.org

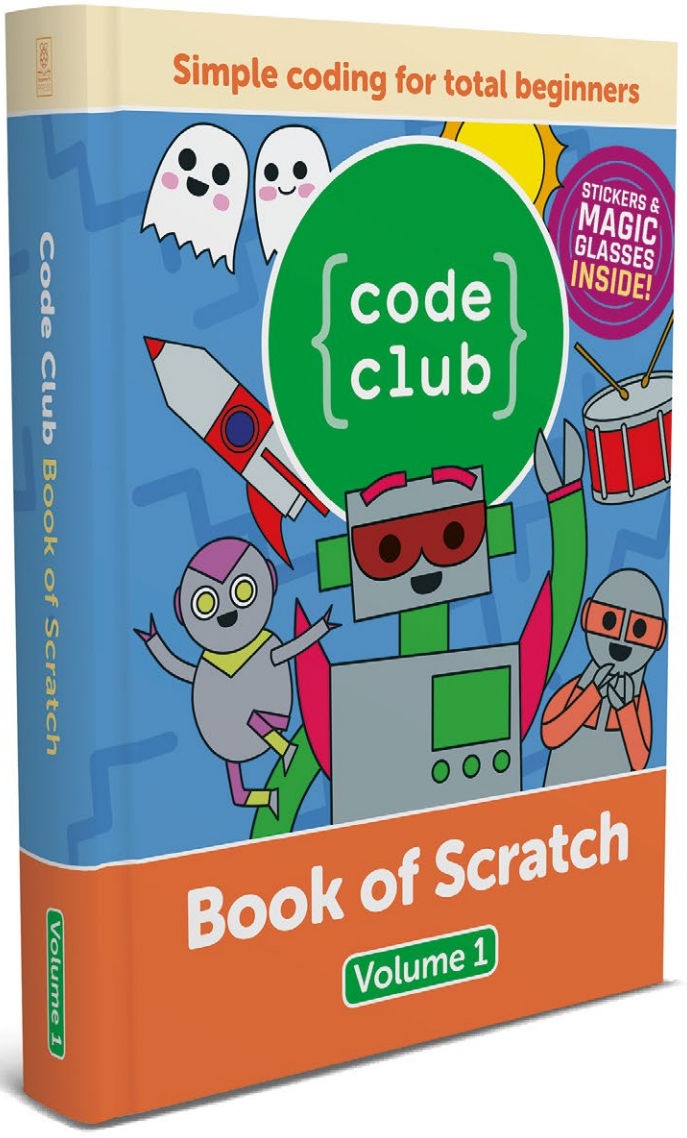
£9.99
Free worldwide shipping!

code club

# Book of Scratch

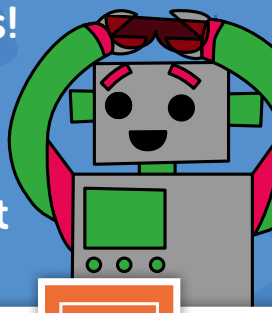
Volume 1

Simple coding for total beginners




## The first Code Club book has arrived!

- Learn to code using Scratch, the block-based language
- Follow step-by-step guides to create games and animations
- Use the magic glasses to reveal secret hints
- Includes 24 exclusive Code Club stickers!
- The special spiral binding allows the book to lay flat

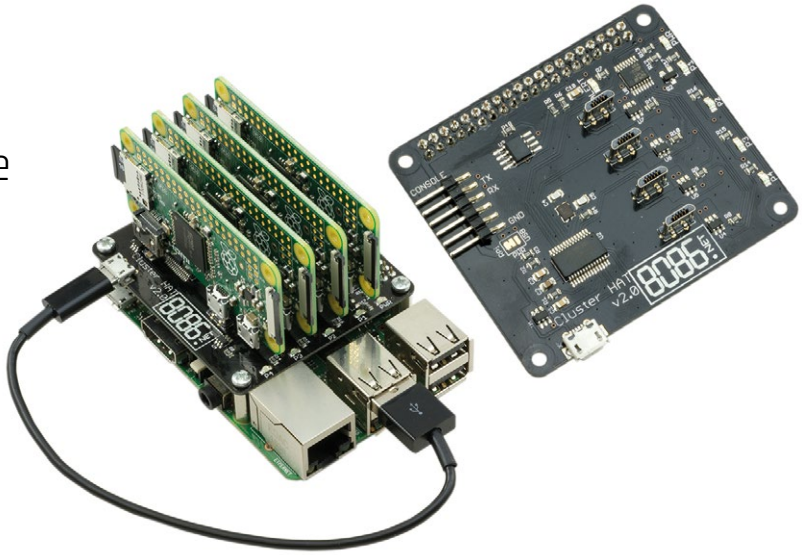


**Available at: magpi.cc/CCbook1**

Raspberry Pi PRESS

# WIN One of Ten Cluster HATs and create a Pi powerhouse!

> The Cluster HAT is a simple and compact way to build a low-cost USB-connected Raspberry Pi Bramble



Take a Raspberry Pi 3B+, add a Cluster HAT, and fill it with up to four Pi Zero boards and you've created a parallel-processing beast that lets you do some really fun stuff.

8086 Consultancy has given us ten Cluster HATs to give away, so enter today for your chance to win one (Pi 3B+ and Pi Zero boards not included).

**Head here to enter:** magpi.cc/win | **Learn more:** clusterhat.com

## Terms & Conditions

Competition opens on **19 December 2018** and closes on **31 January 2019**. Prize is offered to participants worldwide aged 13 or over, except employees of the Raspberry Pi Foundation, the prize supplier, their families or friends. Winners will be notified by email no more than 30 days after the competition closes. By entering the competition, the winner consents to any publicity generated from the competition, in print and online. Participants agree to receive occasional newsletters from The MagPi magazine. We don't like spam: participants' details will remain strictly confidential and won't be shared with third parties. Prizes are non-negotiable and no cash alternative will be offered. Winners will be contacted by email to arrange delivery. Any winners who have not responded 60 days after the initial email is sent will have their prize revoked. This promotion is in no way sponsored, endorsed or administered by, or associated with, Instagram or Facebook.

# HackSpace

TECHNOLOGY IN YOUR HANDS

## THE NEW MAGAZINE FOR THE MODERN MAKER

**SUBSCRIBE AND SAVE UP TO 35%** on the cover price

## ISSUE #14

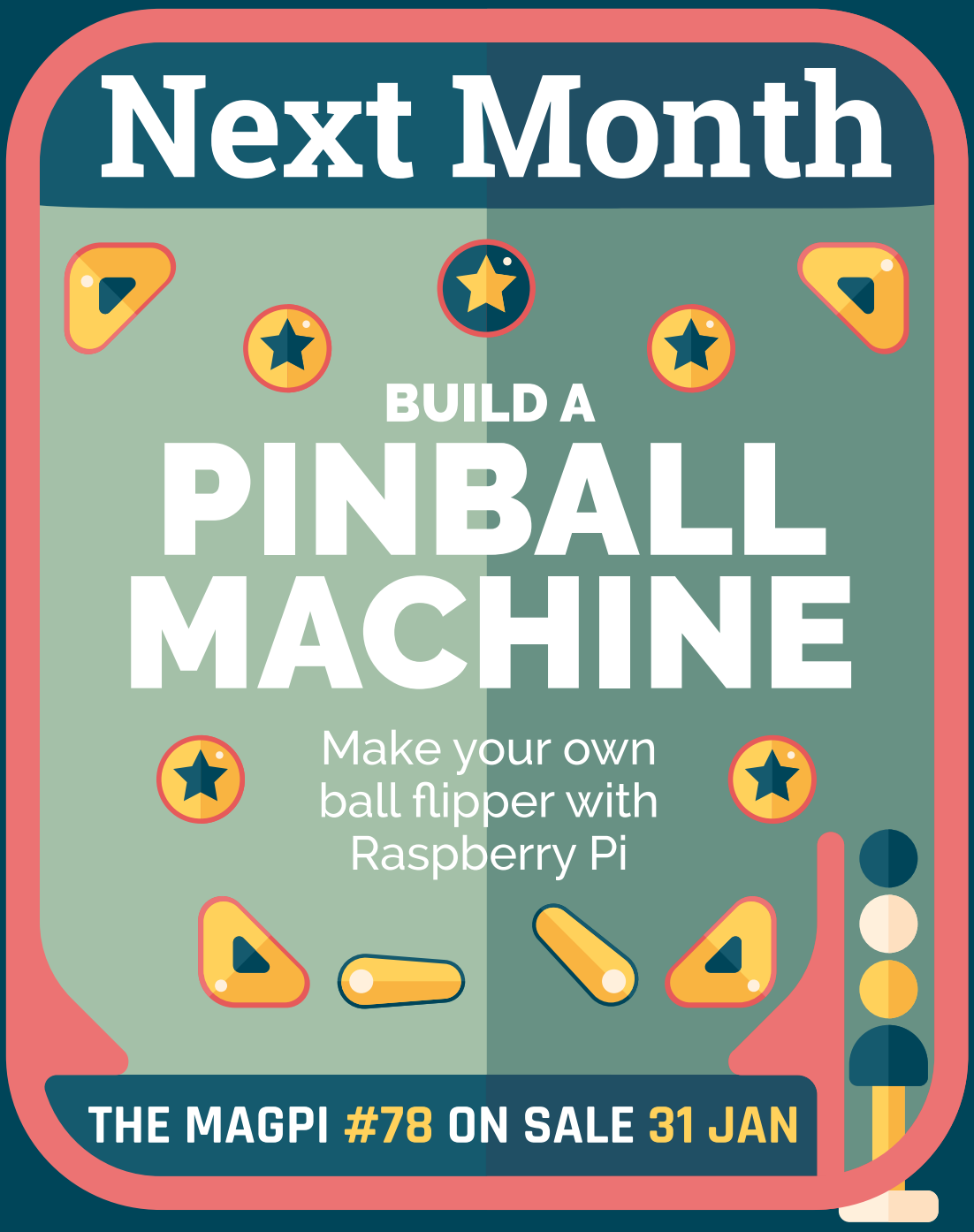
# OUT NOW

hsmag.cc

Available on the App Store

GET IT ON Google Play

# Next Month




## Plus!

**The C64 made from Lego**
Code an isometric adventure game
**Create your own Sense HAT MP3 player**
Build a working telephone exchange

*** Contents subject to change**

DON'T MISS OUT!
**magpi.cc/subscribe**

| | |
|---|---|
| TWITTER | @TheMagPi |
| FACEBOOK | fb.com/MagPiMagazine |
| EMAIL | magpi@raspberrypi.org |


## EDITORIAL

**Editor**
Lucy Hattersley
lucy@raspberrypi.org

**Features Editor**
Rob Zwetsloot
rob.zwetsloot@raspberrypi.org

**Sub Editors**
Phil King and Nicola King

## DESIGN

criticalmedia.co.uk

**Head of Design**
Lee Allen

**Designers**
Mike Kay, Harriet Knight

**Illustrator**
Sam Alder

## CONTRIBUTORS

Brian Beuken, Mike Cook, David Crookes, PJ Evans, Gareth Halfacree, Rosie Hattersley, Nicola King, KG Orphanides, Richard Smedley, Mark Vanstone

## PUBLISHING

**Publishing Director**
Russell Barnes
russell@raspberrypi.org
+44 (0)7904 766523

**Director of Communications**
Liz Upton

**CEO**
Eben Upton

## DISTRIBUTION

Seymour Distribution Ltd
2 East Poultry Ave,
London EC1A 9PT
+44 (0)207 429 4000

## SUBSCRIPTIONS

Raspberry Pi Press
Mann Enterprises, Unit E,
Brocks Business Centre,
Haverhill, CB9 8QP

**To subscribe**
magpi.cc/subscribe

**To get help:**
rpipresshelp@raspberrypi.org

recycle

This magazine is printed on paper sourced from sustainable forests and the printer operates an environmental management system which has been assessed as conforming to ISO 14001.

The MagPi magazine is published by Raspberry Pi (Trading) Ltd., 30 Station Road, Cambridge, CB1 2JH. The publisher, editor, and contributors accept no responsibility in respect of any omissions or errors relating to goods, products, or services referred to or advertised in the magazine.

Except where otherwise noted, content in this magazine is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0).
ISSN: 2051-9982.

# Remember. **Create.**

**K.G. Orphanides** on silence, preservation, and the joy of trashgames

I reviewed the first Raspberry Pi Model B, but it took a while for me to click with it. As Pi iterations became more powerful, their capabilities intersected with the needs of my other projects.

A Pi is at the heart of my silent recording studio for voiceover work and field recording. The biggest problem with home audio recording from a mic input is the inevitable background noise from the fans of a standard desktop or laptop PC.

Equipped with passive heatsinks, there's nothing in my Pi 3 box to make the slightest noise, so I can record almost anywhere without unwanted background noise to cancel out – at the cost of other sounds in the same frequency range – during production.

The next upgrade is a fully portable and ideally weatherproof version for field recording in nature. That's going to call for a touchscreen, a hefty battery, and an IP6X weatherproof chassis.

## Preserving the past

A Pi 3 is also key to my interest in software preservation and emulation. I have a lot of classic computers and consoles, but it's usually my Pi-based emulation box that I get out when I want to show someone what gaming felt like in the 1980s and 1990s. It fits in a pocket and doesn't run the risk of a capacitor going fizz at a crucial moment.

But it's about more than playing well-loved games that are otherwise unlikely to see a re-release. There are whole classes of games and software that are at imminent risk of being lost to history, and the Pi provides a useful test bed for getting a lot of those running, assessed, and documented.

> Write software for the pleasure of puzzling out the logic and structure needed to create a meaningful experience in a few lines of code

While there's no chance that we'll lose Doom, Ultima IV, or Lemmings, preservation is less certain for novelty screensavers, cheap CD-ROM compilations of fascinating Windows 95 'shovelware', 3.5-inch floppies packed with cheat tools and 'trainers' to boost your RPG party's stats, or Java phone games.

I'm using a Pi to find working versions of games like Rovio's Darkest Fear survival-horror/puzzle series – released long before Angry Birds was a hit – and the elusive feature-phone version of Fallout 1, with the intention of curating an online archive.

## Coding the future

But I most love the Pi for the opportunities it holds for new developers of all ages. Despite numerous unfinished projects and false starts, I didn't release my first game – the text adventure 'Eight characters, a number, and a happy ending' – until my mid-30s.

While I don't exclusively develop on a Pi, it's what got me back into programming. It reminded me that I could write software for the joy of it; for the pleasure of puzzling out the logic and structure needed to create a meaningful experience in a few lines of code.

Like home microcomputers of the 1980s, the Pi can turn anyone into a bedroom programmer, ready to join the thrilling indie, art games, and wonderfully glitchy trashgames scenes born of widespread internet access and high-quality, free development tools like Twine and PICO-8.

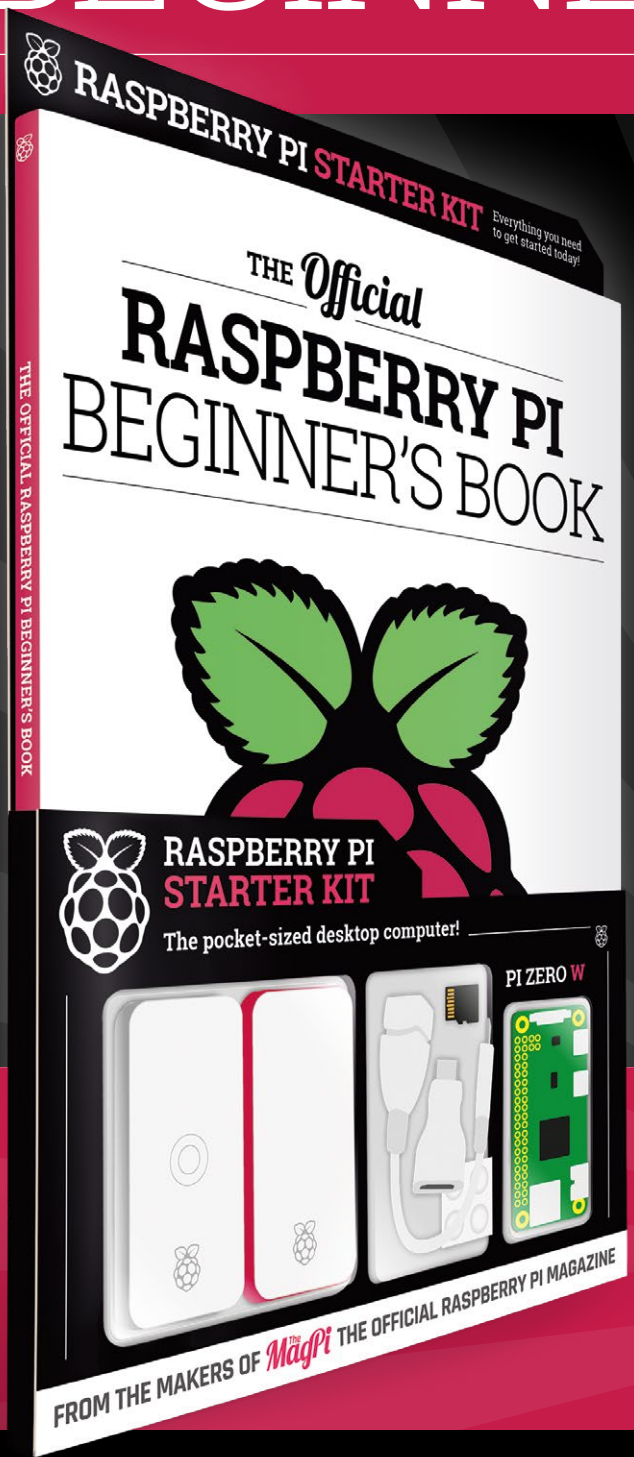
AUTHOR

K.G. Orphanides

K.G. is a writer, developer, and software preservation enthusiast with a penchant for narrative games, nineties CRPGs, and owlbears.

**@KGOrphanides**

# THE *Official* RASPBERRY PI BEGINNER'S BOOK


RASPBERRY PI STARTER KIT
Everything you need to get started today!
THE Official RASPBERRY PI BEGINNER'S BOOK
THE OFFICIAL RASPBERRY PI BEGINNER'S BOOK
RASPBERRY PI STARTER KIT
The pocket-sized desktop computer!
PI ZERO W
FROM THE MAKERS OF The MagPi THE OFFICIAL RASPBERRY PI MAGAZINE

## LEARN COMPUTING THE EASY WAY!

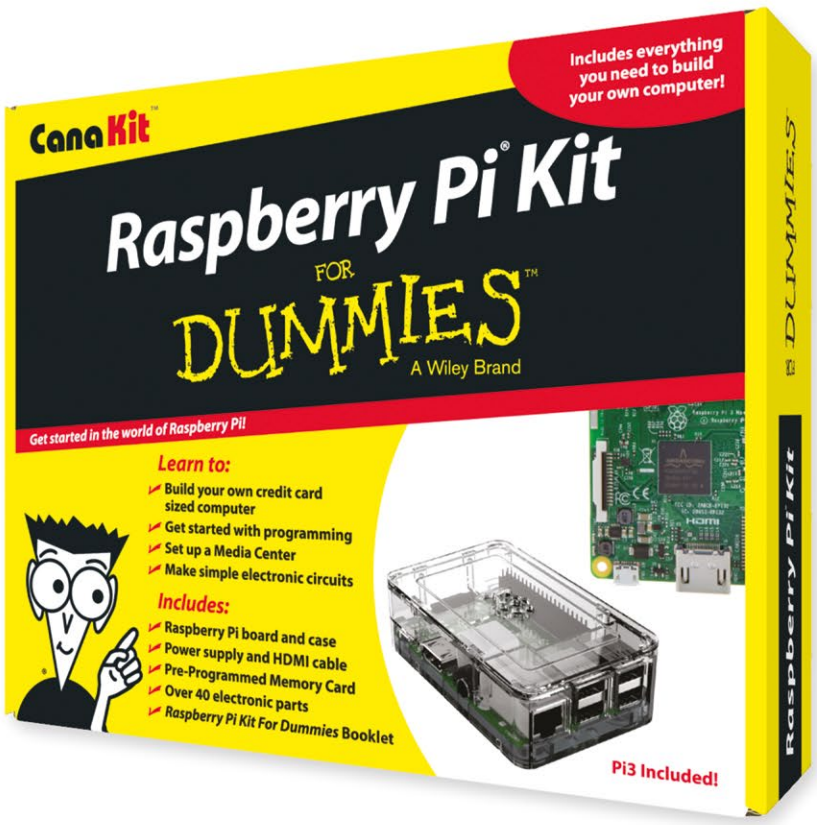
### Includes

- Pi Zero W computer
- Official case with three covers
- USB and HDMI adapters
- 8GB microSD card
- 116-page beginner's book

*Available now*

**Buy online: magpi.cc/store**

CanaKit™


CanaKit
Includes everything you need to build your own computer!
Raspberry Pi® Kit
FOR DUMMIES™
A Wiley Brand
Get started in the world of Raspberry Pi!
Learn to:
Build your own credit card sized computer
Get started with programming
Set up a Media Center
Make simple electronic circuits
Includes:
Raspberry Pi board and case
Power supply and HDMI cable
Pre-Programmed Memory Card
Over 40 electronic parts
Raspberry Pi Kit For Dummies Booklet
Pi3 Included!
Raspberry Pi Kit
FOR DUMMIES

FOR DUMMIES® A Wiley Brand

Available for worldwide shipping at:

**WWW.CANAKIT.COM**

Available in Europe through RS Components

RS

## Kit Includes:

- *Raspberry Pi For Dummies* Booklet
- Raspberry Pi 3 Board
- Memory Card
- Plastic Case
- 2.5A Power Supply
- HDMI Cable
- Resistors
- LEDs
- Push Button Switches
- Prototyping Breadboard
- Jumper Wires
- Heat Sinks

**$89.99** US DOLLARS

**£69.99** EXCLUDING VAT

Raspberry Pi is a registered trademark of the Raspberry Pi Foundation. For Dummies and the Dummies Man logo are trademarks or registered trademarks of John Wiley & Sons, Inc. Used under license. RS logo is a registered trademark of RS Components Ltd. CanaKit is a registered trademark of Cana Kit Corporation.

**ELECTRONIC KITS • ELECTRONIC PARTS • RASPBERRY PI • ARDUINO**